Classification of Symbols handwritten by Children

Master Thesis

Adrian Lars Benjamin Iten
15-722-291

Submitted on
November 8, 2022

Thesis Supervisor
Prof. Dr. Manuel Günther
Prof. Dr. Bernd Gärtner, ETH Zürich

University of Zurich
Department of Informatics

Master Thesis

Author: Adrian Lars Benjamin Iten, adrianlarsbenjamin.iten@uzh.ch

Project period: 09.05.2022 - 09.11.2022

Artificial Intelligence and Machine Learning Group
Department of Informatics, University of Zurich

Acknowledgements

I want to thank everyone who was involved in this project in any way. First, I want to thank the supervising professors, Manuel Günther and Bernd Gärtner, for their continuous support and constructive feedback. Second, I want to thank the whole Herby team for providing their platform and support. In addition, I want to thank everyone who helped in the data collection process. In particular, I want to thank Niklas Frei for his mini-booklet contribution, as well as his crucial role in the distribution of the exercise sheets. I also want to thank all the teachers for including the exercise sheets in their lessons and the students for solving the exercise sheets. Furthermore, I want to thank Vishwas Kalani for labeling the majority of the booklets. Last but not least, I want to thank my family and friends for supporting me in various ways, including the test-wise solving of exercise sheets.

Abstract

Classification of handwritten symbols like digits or letters is well-studied. This master thesis focuses on the novel domain of *Kinderlabor* computer science exercises. It contributes a dataset of symbols handwritten by children in the corresponding domain and evaluates different classification models. This thesis is part of a larger project which aims to implement an automated correction process for those exercises using existing localization and correction algorithms in combination with a symbol classification model developed in this thesis, which classifies each symbol independently. The dataset is collected using different types of exercise sheets, of which the data collected from productive exercise sheets have a significant drawback of lacking or even entirely missing some symbols. To overcome this limitation, a very time-efficient exercise sheet that contains all symbols is contributed. This thesis starts by inspecting the data, where different characteristics of the handwritten symbols and the prevalence of certain symbols are studied. Then, three different data splits are defined, including a data split to assess performance in the productive application scenario, where the model has to classify symbols from new school classes. Two important characteristics of the dataset are the label imbalance and that the dataset contains a certain amount of unknown symbols, making the classification problem an open set classification problem. In an open set classification problem, a classification model must not only correctly classify a set of known symbols, but also reject unknown symbols. Two types of experiments are then performed on the dataset: First, baseline models for correctly classifying all known symbols, including empty fields, are created that are not explicitly trained to reject unknown symbols. Subsequent experiments are performed to evaluate the ability of the models to reject unknown symbols while maintaining good performance on the prediction of known symbols. As existing work lacks an open set evaluation metric for imbalanced datasets, an adaptation to the existing open set classification rate curve is contributed and used throughout the experiments.

Zusammenfassung

Die Klassifizierung handgeschriebener Symbole wie Zahlen oder Buchstaben ist gut erforscht. Diese Masterarbeit beschäftigt sich mit Programmieraufgaben im Rahmen der *Kinderlabor* Organisation. Diese Arbeit trägt einen neuen Datensatz bei, welcher handgeschriebene Symbole von Kindern im Rahmen dieser Aufgaben enthält und evaluiert verschiedene Klassifizierungsmodelle. Sie ist Teil eines übergeordneten Projektes mit dem Ziel, einen automatischen Korrekturprozess für solche Aufgaben zu implementieren auf Basis von existierenden Ortungs- und Korrekturalgorithmen und einem im Rahmen dieser Masterarbeit entwickelten Klassifizierungsmodell, welches jedes Symbol unabhängig klassifiziert. Der Datensatz wird mithilfe von verschiedenen Typen von Arbeitsblättern gesammelt, wobei die Daten der produktiven Arbeitsblätter den Nachteil haben, dass gewisse Symbole nur sehr selten bis gar nicht vorkommen. Um diese Knappheit zu überwinden, trägt diese Arbeit ein zeiteffizientes Arbeitsblatt zur Datensammlung bei, welches alle Symbole enthält. Die Masterarbeit analysiert zuerst den Datensatz, wo verschiedene Aspekte der handgeschriebenen Symbole und deren Häufigkeit untersucht werden. Es werden drei verschiedene Datentrennungen definiert, von welchen eine genutzt werden kann, um die Genauigkeit in der produktiven Benutzung durch neue Schulklassen zu evaluieren. Zwei wichtige Charakteristiken des Datensatzes sind die ungleiche Häufigkeit der Symbole und das Auftreten von unbekannten Symbolen, was das Klassifizierungsproblem zu einem Open Set Klassifizierungsproblem macht. In einem Open Set Klassifizierungsproblem muss ein Modell nicht nur bekannte Symbole richtig klassifizieren, sondern auch unbekannte Symbole zurückweisen. Im Anschluss werden zwei Arten von Experimenten durchgeführt: In einem ersten Schritt werden verschiedene Referenzmodelle für die korrekte Klassifizierung bekannter Symbole, inklusive leeren Feldern, verglichen. Diese Modelle werden nicht explizit darauf trainiert, unbekannte Symbole zurückzuweisen. In einem zweiten Schritt werden Modelle auch explizit darauf trainiert, unbekannte Symbole zurückzuweisen, währenddem sie eine hohe Genauigkeit auf bekannten Symbolen halten sollen. Da in bestehenden Werken keine angemessene Open Set Bewertungsmetrik für unausgeglichene Datensätze existiert, wird in dieser Masterarbeit eine Anpassung der Open Set Klassifizierungsratenkurve beigetragen und in den Experimenten benutzt.

Contents

Chapter 1

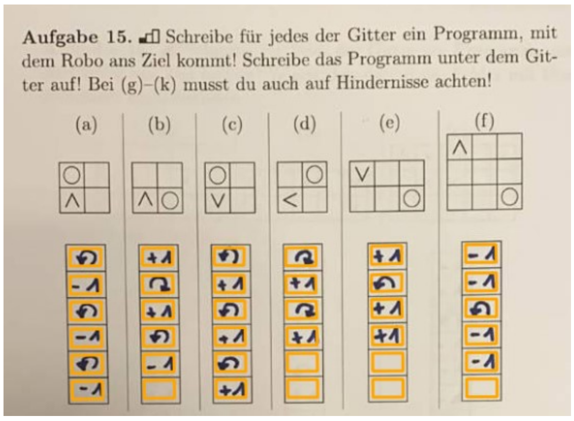
Introduction

Optical Character Recognition (OCR) has evolved rapidly in the past. Current state-of-the-art OCR systems such as Google Docs OCR allow the localization and classification of a large number of written languages, including languages that originate from Latin, Arabic, Chinese or other alphabets Tafti et al. (2016). Many different systems have been built to recognize different domains, such as mathematical expressions or musical symbols. While the first systems were rule-based, the introduction of Deep Learning (DL) models has further improved the performance of OCR systems Wei et al. (2018).

Kinderlabor, an independent Swiss organization that provides teaching materials for children aged 4 to 12, is working on a project with the goal of using OCR technology to correct their computer science exercises. Those computer science exercises can be grouped into three exercise types that differ in the symbols that children need to draw to solve the exercise. The general theme of the exercise is that a robot called *Robo* moves on a grid by executing a sequence of instructions. Figure 1.1 shows an example of the first two exercise types. In the first type of exercise, children need to draw symbols, which represent instructions, to reach a given target after executing these moves on the grid. In the second type of exercise, children need to draw the location of the robot on the grid after it has executed all the instructions starting from a given starting position on the grid. The third type of exercise is simpler as it consists of a user marking a checkbox using an X or leaving it empty. A full list of all valid symbols and some examples of how children write them are provided in the following chapters.

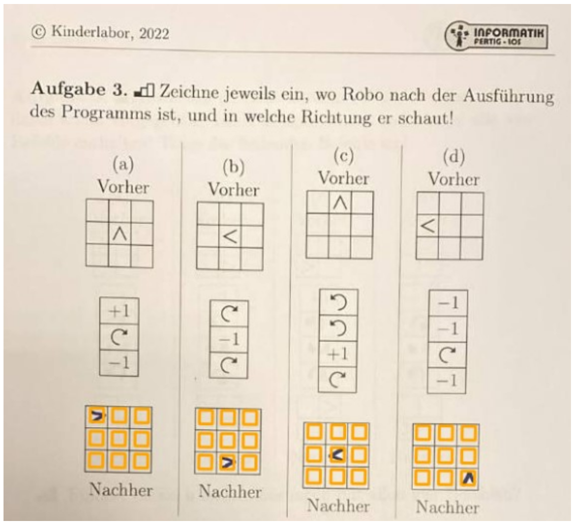
The project foresees that the correction process is implemented using a sequence of three independently developed modules that solve different steps of the correction process. From the perspective of a child, the correction process shall be made as simple as taking a photograph of an exercise sheet and getting the correction drawn into this photograph. First, a software entitled *Herby* is used to identify the sheet in the photograph and localize pre-defined symbol boxes, where the children need to draw the symbols. Subsequently, a classification model is used on each of those boxes independently to classify which symbol is written in this box. It uses the cropped box as input and returns a probability for each symbol, with the objective of the actual symbol having the highest probability. As a third step, a correction algorithm uses these probabilities in combination with the definition of the exercise to generate visual feedback for the child.

This thesis evaluates different DL models that can be used in the correction process to solve the second step of independently classifying symbols handwritten by children. Having no existing labeled data on this domain, this thesis contributes a dataset which is collected from photographs and scans of Kinderlabor exercise sheets solved by children, where the symbol boxes are individually cropped from the original photograph with the location determined by the aforementioned Herby software. As the localization of the symbol using existing Herby software precedes the classification, errors in the localization may propagate to the symbol classification and subsequently to the correction. Hence, this thesis also seeks to mitigate the impact of slight localization



Aufgabe 15. Schreibe für jedes der Gitter ein Programm, mit dem Robo ans Ziel kommt! Schreibe das Programm unter dem Gitter auf! Bei (g)–(k) musst du auch auf Hindernisse achten!

(a) Instruction Exercise Type Example



© Kinderlabor, 2022

Aufgabe 3. Zeichne jeweils ein, wo Robo nach der Ausführung des Programms ist, und in welche Richtung er schaut!

(b) Location Exercise Type Example

Figure 1.1: EXAMPLES OF (A) INSTRUCTION AND (B) LOCATION EXERCISE TYPE. In (a), children are shown a grid and need to instruct Robo to move to the target field (circle) and in (b), children are shown a starting position and a sequence of instructions and need to draw the final location of Robo. The bounding boxes predicted by the preceding Herby localization algorithm are illustrated as orange boxes.

errors in the preceding Herby localization.
Because the collected dataset contains different types of unknown symbols, as explained in more detail in Chapter 3, the classification problem can be viewed as an open set classification problem. In an open set classification problem, the model is not limited to classifying a list of known symbols, but also has to reject unknown symbols. In the foreseen correction process, this information can be used by the correction algorithm in order to provide a better visual feedback stating that a symbol is unknown and cannot be interpreted. This can be seen as a way to explain the concept of syntax errors in programming languages to children.
Based on the problem statement, this thesis addresses the following three research questions (RQ):

- RQ1: How can the dataset be used to assess the model performance for productive use, which is classifying symbols originating from new school classes?
- RQ2: What techniques can be used to maximize model performance and minimize the impact of preceding localization offsets for an independent classification model?
- RQ3: How can the open set performance of a model be assessed for the dataset and can it possibly be improved by using some sort of unknown symbols in training?

The thesis first presents additional background information on the specific domain and DL techniques in Chapter 2. In Chapter 3, the collected dataset for this domain is explained in more detail by showing examples, label distributions, particular observations, and how the data are split into training and test sets. Subsequently, Chapter 4 explains the approaches used in the experiments and particular design choices. Chapter 5 contains the experiments, where different training approaches for DL models and their performances are compared. Chapter 6 discusses the results, lists limitations, and proposes future work. Finally, Chapter 7 gives a conclusion of the thesis.

Chapter 2

Background

This chapter includes background information on the different topics relevant to this work. It starts by providing more details about the real-world application that this project builds around and then explains the technical concepts used in the experiments. The chapter is rounded off with a list of related work.

2.1 Kinderlabor Exercises

The productive Kinderlabor computer science booklet includes open-text exercises, as well as the three different types of code evaluation and coding exercises mentioned in the introduction chapter Gärtner et al. (2021). While open-text exercises are not covered in this thesis, the other three types of exercises each involve a specific alphabet, for which a domain-specific classification model is created. This section provides the set of symbols that children are intended to draw in the respective exercise types and their semantic meanings.

1. **Instruction**: The first type of exercise is to give Robo a sequence of instructions on how to reach its target based on a grid that depicts the starting position of Robo and a target field it is supposed to reach, as well as some obstacles that Robo has to avoid on its way. Robo can be given four basic instructions: move forward by one field ($+1$), move backward by one field (-1), turn clockwise in place (↻), and turn anti-clockwise in place (↺). More advanced exercises introduce (nested) loops of fixed sizes, instructing Robo to execute the code within this loop either twice, three, or four times. As in many programming languages, the notation consists of a loop start symbol ($2^{\ulcorner}$, $3^{\ulcorner}$, $4^{\ulcorner}$) and the end-of-loop symbol ($\lrcorner$) that enclose the sequence of code to be executed multiple times. If the program is shorter than the maximum number of instructions allowed, an instruction symbol box can be left empty.

2. **Orientation**: The second type of exercise covers the question *At which field does Robo end up after executing a given sequence of instructions from the starting point and what direction is it looking at?* In this type of exercise, the exercise contains a grid with Robos starting position as well as a sequence of instructions, and the children are then supposed to draw the final location of Robo, as well as its orientation on the grid. Thus, there are five different states of each field: The first state is that this field is empty because Robo is not located in this field after executing the sequence of instructions. The other four states are drawn as arrow tips that point in the respective direction into which Robo is looking as it is located on this field after finishing the execution of all the instructions. For example, $>$ represents that Robo is located in the respective field and looking to the right. Looking up ($\wedge$), left ($<$) and down ($\vee$) are other valid symbols that a child can draw into such a field.

3. **Checkbox**: The third type of exercise covers binary questions like *Can Robo execute this code without crashing into an obstacle?* or *Can Robo find a way from the starting point to the endpoint?* This type of exercise shall be responded to by the children with *yes* by checking the box with a cross (X) or with *no* by leaving the box empty.

This thesis uses a slightly different notation in tables and figures to simplify the distinction and interpretation of symbols. In particular, the notation $2x$, $3x$ and $4x$ is used for the loop start symbols and *end* is used for the end-of-loop symbol. The orientation symbols are represented by full arrows ($\uparrow$, $\rightarrow$, $\downarrow$, $\leftarrow$) and an empty symbol box is denoted *empty*.

2.2 Herby

As explained in the introduction, Herby is a closed-source software solution, which was developed by a startup named the same Pelican (2019). The software can be used to correct exercise sheets based on a photograph of the exercise sheet solved by a student. Internally, the software recognizes the corresponding document from a document pool and aligns the exercise sheet. This allows the localization of pre-defined boxes, where a child has to write their answer, and each box can then be cropped in an orientation-corrected manner if the original photograph was taken rotated or similar. However, the localization algorithm and its rotation correction do not offer perfect accuracy and therefore deviations from the predicted bounding box to the actual location of the bounding box are possible.
Figure 2.1 shows the third step of the correction process, for which Herby provides a toolbox of existing functionality: Herby allows the visualization of feedback directly projected into the photograph of an exercise sheet by grouping symbol boxes and drawing feedback over a group of one or more boxes. The implementation of the correction algorithm is not part of this thesis; however, potential limitations introduced by the classification model that can propagate to the correction algorithm are considered in the discussion.

2.3 Deep Learning

Deep learning (DL) is a machine learning technique that learns a parameterized differentiable function. DL solutions have shown to be superior to previous machine learning algorithms on different types of approaches, including computer vision tasks like image processing LeCun et al. (2015).
DL models, also known as artificial neural networks, usually consist of multiple layers. Each layer transforms an input signal in the form of a tensor and returns an output tensor. The simplest example is a linear layer that applies a linear function of the form $Ax + b$ to the input signal x. Each layer can have trainable parameters, which are called weights. For a linear function of the form $Ax + b$, those weights are the matrix A and the bias b. Other layers include nonlinear activation functions such as Rectified Linear Units (ReLU), which apply the element-wise function $ReLU(x) = max(0, x)$. ReLU can also be parameterized so that the function multiplies negative values with a trainable parameter, which is then referred to as PReLU He et al. (2015). Furthermore, different regularization layers exist, like dropout that randomly drops units with a given probability Srivastava et al. (2014). A very important type of layer for the task of image processing in particular are convolution layers. As an example, a 2D convolution with a kernel size of 3x3 iterates over the pixels of a grayscale image and applies a dot product of a 3x3 weight matrix with the corresponding pixels of the 3x3 excerpt of the image. The convolution is applied to the whole image in a sliding window, moving by at least 1 pixel in width and height. The number of pixels that the convolution moves in the sliding window is called stride and additionally, padding

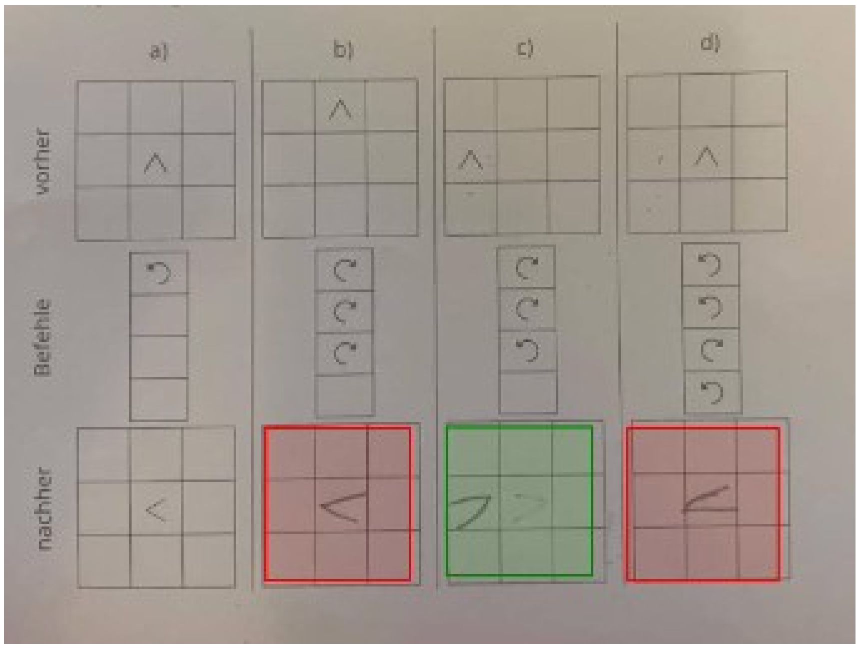


Figure 2.1: EXAMPLE OF HERBY CORRECTION ON AN EXERCISE SHEET. A red box is drawn over a wrong answer grid and a green box is drawn over a correct answer grid.

may be used at the border of the image. Usually, every convolution layer operates on a number of input channels as channeled convolution, allowing it to detect shapes within multiple input channels like the three color channels red, green and blue in RGB images. Another important type of layer in image processing are pooling layers. As an example, 2D max pooling operates on the image in a sliding window, summarizing the features in the region by returning the maximum value. Like convolution layers, pooling layers can also be configured by changing the stride or by adding padding. The pooling layers treat each input channel independently and traditionally do not contain trainable parameters. Pooling can also be performed globally; for example, global max pooling can be used to reduce the entire channel to a single number. After applying a number of layers, the model returns an output which represents the desired output, for example, the probabilities of different image classes for the task of image classification.

The second important building block of DL is a loss function that is calculated based on the target and the actual model output. Cross Entropy (CE) loss is a popular loss function used in classification tasks to minimize cross entropy between a target label and the model output. It can also be further adjusted by applying label smoothing Szegedy et al. (2016).

The last building block of DL are optimization algorithms. DL models are usually trained in an iterative manner, updating the weights based on a fixed-size subset of input samples referred to as mini-batch or batch. In the training loop, first, the output and the current loss of the model for a batch is computed. Subsequently, the gradient of the loss with respect to the current weights is computed. Lastly, the weights are updated to reduce the loss using gradient-based optimization algorithms like (stochastic) gradient descent. After performing enough model update steps, the model eventually converges toward a certain (local) minimum of the loss function, having found the optimal parameters for the task. Update steps may also be measured in epochs, which corresponds to a full iteration over all batches of the training set.

When a model is trained on a target originating from labeled data, such as the class of an image, this is referred to as supervised learning. When no such labels are available, this is usually re-

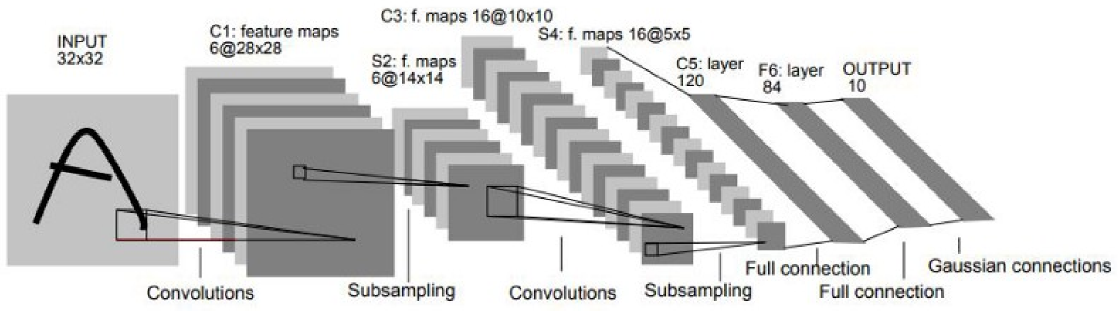


Figure 2.2: ORIGINAL LENET-5 ARCHITECTURE. The illustration is from the original authors LeCun et al. (1998). In this thesis, slight adaptations are made to the network, for example, using a full connection instead of Gaussian connections. Further differences exist in the use of activation functions and how the subsampling is performed.

ferred to as unsupervised learning. Having explicit labels in the dataset presented in Chapter 3, this thesis is limited to supervised learning.

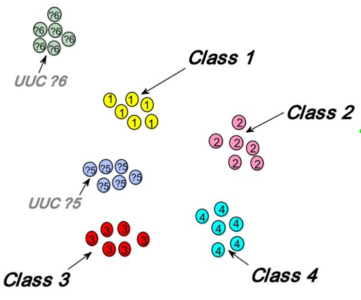
2.3.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a type of artificial neural network that is typically used in image processing, but they can also be adapted to work in other problem domains as well. Like every other DL model, CNNs consist of multiple layers but mainly rely on convolution layers to extract meaningful optical patterns or shapes from the image to generate a deep representation Albawi et al. (2017). In image classification, the deep representation is usually flattened[1] and then a series of fully connected layers is applied to perform the classification.

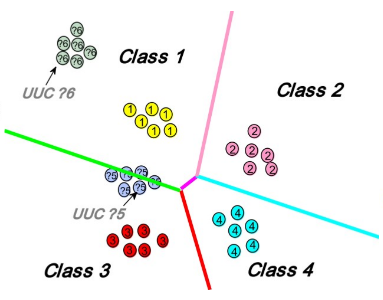
LeNet-5

One of the first CNNs that has become very popular in the research community is the LeNet-5 model LeCun et al. (1998). Its architecture is relatively simple. The model consists of seven computational layers and operates on a 32x32 grayscale input. The first layer, which is denoted as C1 in Figure 2.2, applies a 5x5 channeled convolution that results in 6 channels. The subsequent S2 subsampling layer reduces the size of each channel to 14x14. Subsequently, layer C3 applies another channeled convolution of size 5x5 with a larger size of 16 channels, followed by subsampling S4 that again reduces the size to 5x5. The resulting feature maps are then flattened and followed by fully connected linear layers C5 and F6 that apply a linear transformation that transforms the representation into a vector of size 120, respectively, 84. To obtain the classification, the original model then applies a radial basis function to predict the probabilities of the labels, denoted as Gaussian connections in Figure 2.2. The implementation in this thesis uses a fully connected layer instead. The original LeNet-5 implementation also contains activation functions applied after the convolution layers and linear layers. The adapted version in this thesis uses ReLu after the convolutions and no activation functions after the linear layers. Furthermore, the implementation in this thesis uses 2x2 max pooling to perform the subsampling.

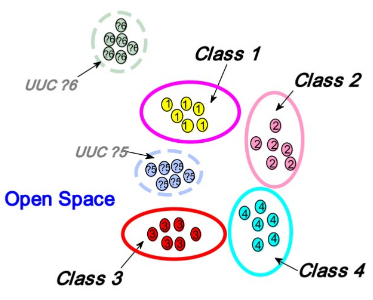
[1]Flattening refers to reshaping tensor dimensions to a 1D vector



(a) Original Dataset Distribution



(b) Closed Set Decision Boundaries



(c) Open Set Decision Boundaries

Figure 2.3: Example of (c) Open Set vs (b) Closed Set Model Decision Boundaries on (a) a given Dataset. The figure is from the survey paper Geng et al. (2020). There are two types of unknown samples (UUC) present in the dataset. A closed set model creates decision boundaries that split the whole space into five sub-spaces while an open set model aims to generate decision boundaries that leave room for unknown samples in the open space.

SimpleNet

In recent years, wider and deeper networks have outperformed simple networks like the aforementioned LeNet-5 on a number of image classification benchmarks. As an example, the 22-layer deep GoogleNet architecture was presented, which consists of multiple inception modules. Such an inception module combines convolutions of different sizes, as well as dimension reduction using max pooling Szegedy et al. (2015). Another popular approach was to use residual nets which enable a deeper model to combine the output of a function with its input by using so-called residual functions He et al. (2016).
SimpleNet is a deep CNN that follows the paradigm suggested by the paper's title *Let's keep it simple, using simple Architectures to outperform deeper and more complex Architectures* Hasanpour et al. (2016). The architecture of the SimpleNet model consists of multiple convolution blocks. Each convolution block applies a relatively simple (3x3 or 1x1) 2D convolution that is followed by batch normalization[2] and a ReLU activation function. After some of the aforementioned convolution blocks, 2x2 max pooling with a stride of 2x2 and dropout regularization is applied. The model then applies global max pooling after the last convolution layer before applying a linear layer and returning the model output. The authors also propose a method to slim the model by reducing the number of channels used in the convolution layers.

2.4 Open Set Classification

Traditionally, classification problems are limited to a set of known labels, and the goal is to classify an input into one of those labels. However, reality can be different as unknown samples that do not belong to any of the known labels can occur. For example, in the case of Kinderlabor exercises, a child can draw a variation of a known symbol or even a completely unknown symbol in the bounding box. Examples of unknown symbols are provided in the following chapter in Section 3.4.
Open set classification is visualized in Figure 2.3, where a dataset is shown in a 2D space. A closed set classification approach splits the whole 2D space into the known labels, while an open set classification approach generates boundaries that leave an open space for unknown samples.

[2]A regularization technique that rescales the values of each channel to have a trainable mean and standard deviation

Source	Domain	Dataset
Ramadhan et al. (2016)	Math	Mouchère et al. (2016)
Lee et al. (2016)	Music	Calvo-Zaragoza and Oncina (2014)
Cireşan and Meier (2015)	Chinese	Liu et al. (2013)
Sonawane and Shelke (2018)	Devanagari Script	own
Dhamija et al. (2018)	Various	Deng (2012), Netzer et al. (2011), more
Miller et al. (2021)	Various	Deng (2012), Netzer et al. (2011), more
Suter (2022)	Various	Deng et al. (2009)
Anacona et al. (2022)	Various	Deng et al. (2009)

Table 2.1: COMPARISON OF RELATED WORK ON DOMAINS AND DATASETS. The related work in open set classification is not limited to a specific domain, using either a variation of different datasets or a set of selected classes from the ImageNet dataset, which is not limited to a specific domain.

Source	Model Approach	Open Set Approach
Ramadhan et al. (2016)	CNN	-
Lee et al. (2016)	CNN	-
Cireşan and Meier (2015)	CNN	-
Sonawane and Shelke (2018)	CNN	-
Dhamija et al. (2018)	CNN	Novel Loss Functions & Evaluation
Miller et al. (2021)	CNN	Novel Loss Function
Suter (2022)	CNN	Comparison of Techniques
Anacona et al. (2022)	CNN	Large Scale & Evaluation Metric

Table 2.2: COMPARISON OF RELATED WORK ON MODEL AND OPEN SET APPROACH. All entries use CNNs as the model approach, which emphasizes the superiority of those networks over different model approaches.

In open set classification, some types of unknown samples may be seen during training, and other types of unknown samples from the open space may occur in testing Geng et al. (2020).
Different techniques to build models for open set classification are lined out in the respective survey paper: A straightforward solution is to add a background class for unknown samples during training. Another approach entitled OpenMax Wen et al. (2016) is one of the first distance-based approaches to detect unknown samples, and more complex solutions, which do not require unknown samples in training, have evolved since like class anchor clustering that learns a center per label Miller et al. (2021). Another approach that does require some unknown samples in the training phase is adapting the cross entropy loss function and turning it into the entropic open set or the objectosphere loss, where the loss function is adapted to generate low probability scores for each label when encountering an unknown sample Dhamija et al. (2018).

2.5 Related Work

The image classification task in general and the classification of handwritten symbols specifically is well researched, as shown in Tables 2.1 and 2.2. Existing work is mostly limited in the sense that it builds on a previously published dataset or only does closed set classification. Generally, most recent papers use a variant of deep CNNs as the model approach for the different domains. In the math domain, Mouchère et al. (2016) published a dataset of online handwritten mathematical expressions. In the domain of music, Calvo-Zaragoza and Oncina (2014) published the *homus* dataset for the classification of pen-based music symbol notation. Another domain is the classification of language alphabets. For the language of Chinese, Liu et al. (2013) published a dataset for online and offline handwritten Chinese character classification. Similarly, Sonawane and Shelke (2018) published a dataset for the classification of handwritten Devanagari script characters. One of the earliest and still most popular handwritten symbol datasets is the MNIST handwritten digit classification dataset by Deng (2012).

As the publication of a dataset enables different researchers to evaluate and propose different model approaches, multiple papers have been published on the aforementioned datasets. For mathematical expressions, Ramadhan et al. (2016) provide a baseline using a simple CNN architecture. In the same way, Lee et al. (2016) provide a CNN baseline for the music dataset by comparing different CNN architectures, where the GoogleNet by Szegedy et al. (2015) achieves the best performance. For the Devanagari script dataset, the authors themselves provide a baseline by applying transfer learning[3] to the AlexNet architecture by Krizhevsky et al. (2017). Cireşan and Meier (2015) provide a baseline for the Chinese character classification task by using what they call *multi-column deep neural networks*, which is a technique that combines multiple deep neural networks in a single network by averaging their features.

Related work in the direction of open set classification is not limited to a specific domain. Dhamija et al. (2018) propose the aforementioned novel loss functions and evaluate them in different domains using a proposed evaluation metric entitled *Open Set Classification Rate (OSCR) curve*, which is explained in more detail in Section 4.3.2. Their experiments include the aforementioned handwritten digits, but also include domains beyond handwriting, such as house numbers from Google Street View in the SVHN dataset Netzer et al. (2011). Similarly, Miller et al. (2021) propose a different optimization approach entitled class anchor clustering that can be used in conjunction with any deep neural network and also evaluate this approach on a number of different datasets including the MNIST digits and SVHN. Due to the different datasets and evaluation metrics used, it is hard to compare different open set techniques directly, which is addressed by Suter (2022), where different open set techniques are compared on protocols of different difficulties. Each of the protocols uses different labels of the ImageNet dataset by Deng et al. (2009). In order to assess the applicability on real-world problems, Anacona et al. (2022) provide large-scale open set classification protocols for ImageNet as well as a novel validation metric that addresses the open set goal of both classifying known samples correctly and rejecting unknown samples.

The first contribution of this thesis is that it provides a dataset for a novel domain of handwritten symbols. Further, it provides a baseline for both closed and open set classification by applying a selection of models and open set approaches from related work. Due to the lack of an existing metric to evaluate open set performance on imbalanced datasets, an adaptation to the OSCR curve is contributed to account for imbalanced datasets.

[3] A technique to transfer model knowledge from one domain to a related domain

Chapter 3

Dataset

This chapter explains the data collection process and provides observations made on the collected dataset. It starts with the methodologies, where a specific sheet is contributed to efficiently collect data. Subsequently, this chapter provides some statistics on the dataset and proposes how the dataset can be split to simulate the real-world application. Finally, some examples illustrate particular challenges associated with the dataset.

3.1 Process

The data collection used throughout the thesis was created with the intention of emulating the productive use of the models as closely as possible while overcoming the limitations of data availability. Generally, the process was that different types of exercise sheets were given to a number of participating primary school teachers, who then let their students solve those exercise sheets and returned them to the authors. The different types of exercise sheets are explained in the following section. For logistic reasons, some exercise sheets were returned digitally by scanning them. The authors then used the Herby software to extract the predicted locations of the symbol boxes and two annotators then labeled each symbol box independently within the Herby software. To avoid inconsistencies in the labeling process, specific labeling guidelines were created that are listed in the appendix in Section A.1.1. Additionally, the master annotator revised all the labels of both annotators to enforce a consistent understanding of the labeling guidelines. Herby software was then used to extract the dataset. The dataset contains the cropped symbol boxes[1] as well as a comma-separated values (CSV) file with the target label and additional meta-information about the cropped symbol. The dataset contains the information listed, but some of this information is anonymized in the process due to legal concerns. Bold items represent columns that are made available in the public dataset CSV.

ID	A unique ID of the cropped symbol
Photograph ID	A unique ID of the photograph in Herby software. Each photograph contains a single exercise sheet and can originate from either a scan or a photograph of the exercise sheet.
School Class	A pseudonymous identifier of the school class (for example Cls01)
Student	A pseudonymous identifier of the student who completed the exercise (for example Stud01)
Sheet	The identifier of the exercise sheet (for example BookletP1)

[1]Each symbol box is provided as a JPEG image with the filename *ID*.jpeg

Task Type The type of exercise associated with the symbol (see Section 2.1)

Exercise An identifier of the exercise to which this symbol belongs (for example 1a)

Label The target label of the symbol

3.2 Exercise Sheet Types

During the data collection phase, three different types of exercise sheets were used to collect data of symbols handwritten by children. The first type is the productive booklet provided by the Kinderlabor organization, the second type is a specific exercise sheet that this thesis provides in order to efficiently collect balanced training data and the last type is a drastically reduced mini-booklet, which was created in cooperation with an associated teacher in order to collect more testing data from booklets.

3.2.1 Booklet

In order to work with Herby localization, the Kinderlabor exercise booklet was updated to contain pre-defined symbol boxes. Using previously solved booklets or scans of those would require manual localization, which is time-consuming and would differ in the sense that no such symbol box would be present around the symbols. As a consequence, the data from the productive booklets with symbol boxes were collected from scratch. During the process, it turned out that the whole process including the distribution of the exercise sheets to the teacher is rather time-consuming, as working through the whole booklet requires many lessons, which the teachers need to fit into their teaching schedule. The benefit of this approach is that data are collected in the exact same manner that the correction system including the classification models shall be used in production, hence no deviation from the productive use is to be expected in these data. A particular drawback is that some symbols are heavily underrepresented in the data collected using this type of exercise sheet, as the teachers and their students only managed to work through the first half of the booklet, which does not yet contain some of the symbols like the loop notation and therefore those symbols are heavily underrepresented or even completely absent. As a test user and reference, an associated teacher also solved the complete booklet.

3.2.2 Sheet

As a more time-efficient approach compared to the booklets, a specific exercise sheet with the symbols of the Kinderlabor exercises was created, where the students have to continue drawing the symbols in a given scheme. An example of a sheet filled out by the authors is provided in the appendix in Figure A.1. Using such a sheet, where the symbols are written in a different exercise context, data can be collected in a way such that the labels of the symbols are balanced and should be very close to the way they are written in the booklets. Additionally, this approach is faster because it requires a maximum of one lesson and no additional teacher input. As a particular drawback, it may occur that the symbols differ slightly due to specifics of the sheet, such as smaller boxes, the two-dimensional symbol grid, or students not receiving the same level of explanations and support from the teacher as in the booklet approach.

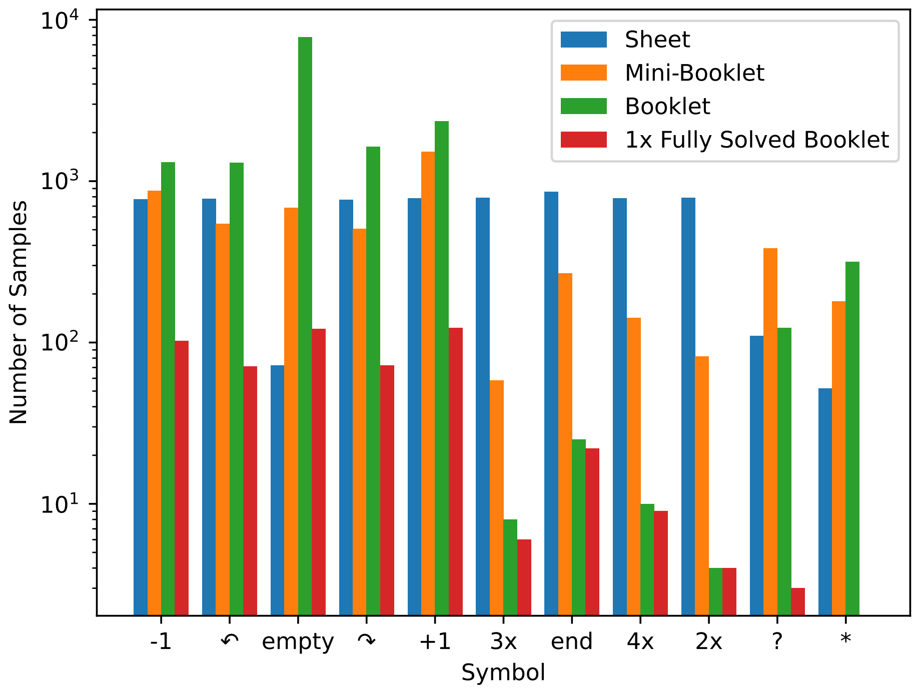


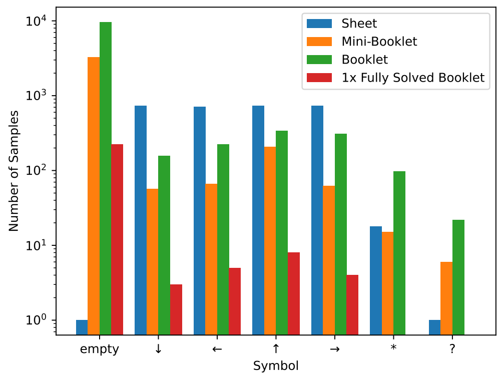
Figure 3.1: Label Distribution of Instruction Symbols. The blue, orange and green bars represent the total number of labeled samples for each label in the respective approach on a logarithmic scale. The question mark represents samples labeled as unknown and the asterisk represents samples marked for manual inspection. The red bar represents a single booklet completely solved by an associated teacher and hence is a subset of the green bar. It serves to show the magnitude of how many samples of each label are to be expected in a fully solved booklet.

3.2.3 Mini-Booklet

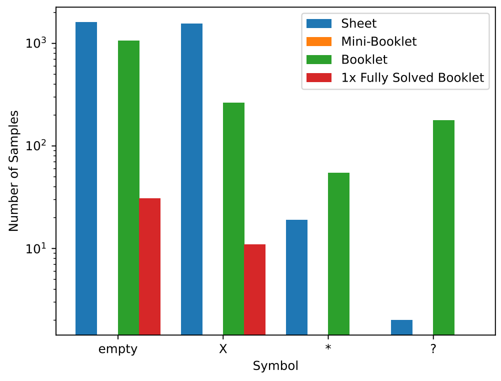
As a second time-saving alternative to the booklet approach, a mini-booklet was created in cooperation with an associated teacher. It contains both instruction tasks and orientation tasks, but no checkbox tasks. It is designed to take fewer lessons than the original booklet and be self-explanatory so that children can work on it independently. It consists of a total of eight tasks on five pages, including both basic and advanced instruction tasks as well as orientation tasks. Its degree of difficulty is higher than the aforementioned specific exercise sheet but lower than the advanced exercises in the booklet. It differs from the specific exercise sheet approach in the way that the exercises build on the true semantic meaning of the symbols. It also differs slightly from the booklet, as the children do not receive an in-depth explanation and discovery of the logic by using building blocks and other additional material provided along with the booklet.

3.2.4 Label Distributions

The dataset contains a total of more than 46’000 labeled samples, including a label for unknown symbols (?) and a label for cases to be inspected manually (*), which are not considered in the ex-



(a) Orientation Symbols



(b) Checkbox Symbols

Figure 3.2: Label Distributions of (a) Orientation Symbols and (b) Checkbox Symbols. The Mini-Booklet does not contain any checkbox exercises, hence no orange bar is visible in (b). Again, the question mark represents unknown symbols and the asterisk represents samples marked for manual inspection.

periments. Additionally, an associated teacher solved the booklet completely as a reference. The distribution of labels differs for both the task type considered and the data collection approach used.

Figure 3.1 shows the distribution of labels for the instruction task and illustrates that the labels are well-balanced for the sheet approach but highly unbalanced for the booklet approach. This can be explained by the fact that a lot of school classes involved in the data collection phase could not finish the booklet and the advanced exercises that use the loop symbols are meant to be solved last due to their increased level of difficulty. The most extreme example is the 2x loop symbol, which is only present within the sheets and the mini-booklet. While the associated teacher used it in their booklet, no child used it in their booklet.

The distribution of the orientation task in Figure 3.2 shows that, due to the nature of this type of exercise, a large number of empty symbol boxes are expected in the booklet, as the child must decide on a single location where Robo ends up on a grid of a fixed size. The four arrow tips are not perfectly balanced but are within the same order of magnitude. For the checkbox task, the label distribution is very similar to that of the orientation task. Due to some single-choice exercises, there are more empty symbol boxes than crosses.

3.3 Data Splits

To take into account different scenarios, three different data splits S1, S2 and S3 are created. These form the basis for the evaluation protocols used in the experiments. Table 3.1 lists all the school classes present in the dataset and how many exercise sheets of each type they contribute to the dataset. Of the 12 school classes, the exercise sheets of four school classes were scanned, and the remaining ones were photographed using a mobile phone, which corresponds to productive use. The first school class (00) includes the associated teacher that filled out the complete booklet and relatives of the author that filled out the sheet to test the sheet methodology. For each data split, first, a training and test set is defined. The validation set is then selected by taking a random 15%

School Class	Capture	S2	Students	Sheet Pages	Mini-Booklet Pages	Booklet Pages
00	Photo	train/valid	5	4	0	21
01	Scan	train/valid	8	8	0	0
02	Scan	train/valid	15	15	0	0
03	Scan	test	18	12	71	0
04	Photo	test	42	0	0	554
05	Photo	train/valid	18	0	0	103
06	Photo	test	7	0	0	45
07	Photo	train/valid	22	22	0	0
08	Photo	train/valid	17	0	101	0
09	Scan	test	16	0	96	0
10	Scan	train/valid	11	0	65	0
11	Photo	test	12	0	56	0

Table 3.1: LIST OF SCHOOL CLASSES IN DATASET. Only data split S2 uses the school class information to split samples. Data split S1 splits the samples based on the data collection methodology (corresponding to the last three columns) and data split S3 splits the samples randomly.

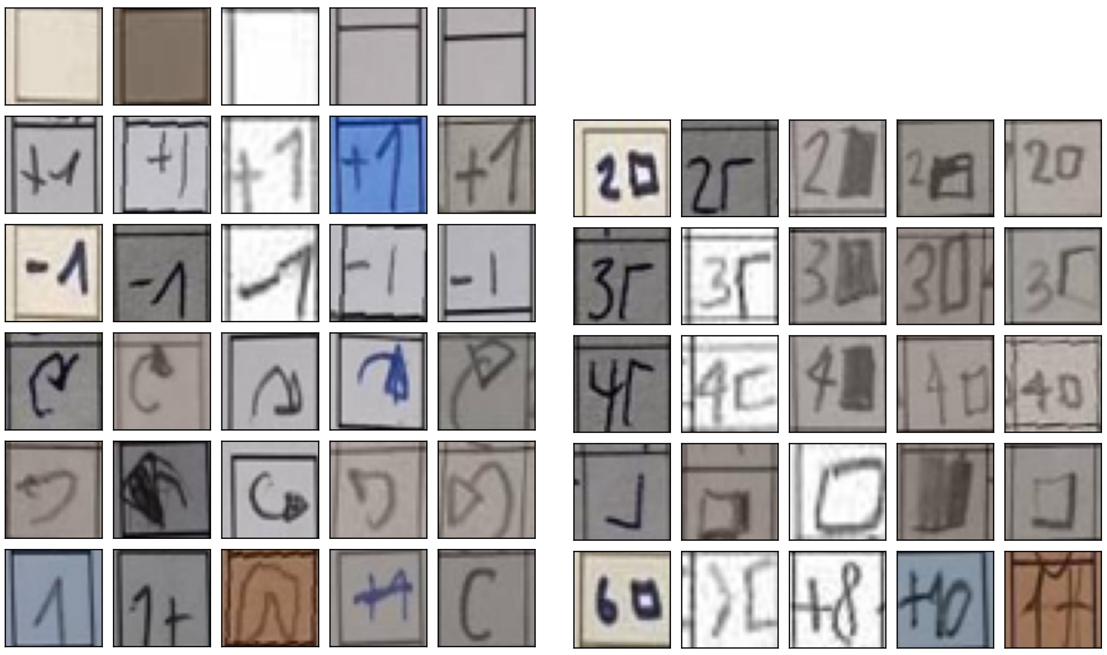
fraction of the training set. Therefore, the model is trained on the remaining 85% of the training set, and the validation set is used to select the hyperparameters and the best-performing model using the validation metric. The following enumeration explains the three data splits in more detail:

S1 Data split S1 splits the data into a training and a test set in such a way that the training set contains all samples that were collected using the previously explained sheet methodology. On the other hand, the test set contains all the samples that come from the booklet and mini-booklet approach, including the booklet completely filled out by a teacher. This data split serves to evaluate whether the sheet methodology is suited for data collection, which is considered to be true if a model trained purely on samples from the sheet methodology achieves good classification performance on the classification of symbols from the booklet and mini-booklet methodologies.

S2 Data split S2 serves to evaluate the classification performance of a model in the productive application scenario, where it has to classify samples from new children, respectively, new school classes that were not seen during training. Table 3.1 lists the 5 of 12 school classes that were selected for the test set with the intention of using most of the productive booklet data in the test set.

S3 Data split S3 splits the symbols randomly into a training and validation set consisting of 80% of the samples and a test set consisting of 20% of the samples. In this split, the training set, before splitting off the validation set, is limited to a maximum of 1′500 samples per label. This data split serves to find an upper bound that may be achieved by a model having data originating from each school class and each child in the training set. Also, the label imbalance in the training set is the smallest for this data split due to the upper limit of training samples per label, which is, however, not reached by the arrow tip symbols in the orientation task type and the advanced instruction symbols in the instruction task type.

Section A.1.2 in the appendix shows the exact number of samples per data split and task type for the training, validation and test sets. Generally, all test sets are imbalanced as some labels provide many more samples than others. For example, there are 9087 empty symbols for the orientation task in the test set of data split S2, and for the arrow tip down symbol, there are only 292 samples in the test set of data split S2. All data splits and evaluation protocols that use them exclude the



(a) Basic Instruction Symbol Examples (b) Advanced Instruction Symbol Examples

Figure 3.3: EXAMPLES OF (A) BASIC AND (B) ADVANCED INSTRUCTION SYMBOLS. The meaning of the symbols is described in Section 2.1. The last row contains unknown symbols.

inspect label as this label includes severe localization errors, which are not analyzed in this thesis. For each data split, a column is added to the public dataset CSV file to indicate whether it belongs to the training, validation, or test set.

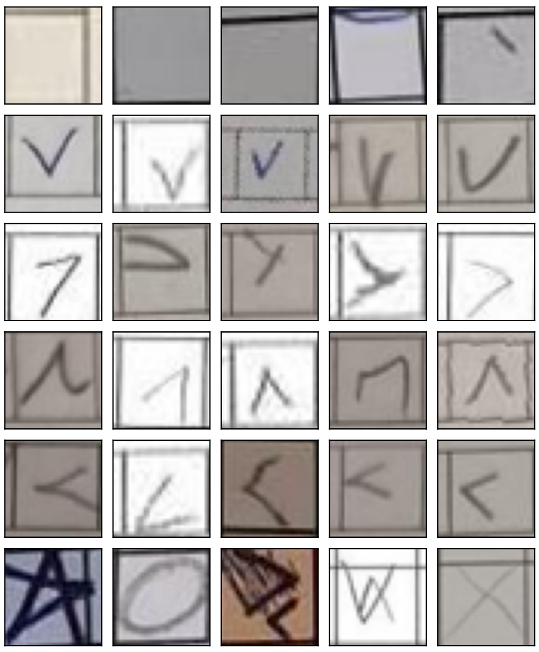
3.4 Observations

By visualizing five training samples per label and task type in Figures 3.3 and 3.4, differences can be observed in both writing styles and lighting conditions. While the differences in the lighting conditions apply globally to all the symbols and task types, some differences in the writing styles only apply to specific symbols. Additionally, each figure contains a row of unknown samples for the respective task type. All the symbols are rescaled to 32x32 despite the original aspect ratio deviating in the specific case of instruction boxes in both the mini-booklet and the booklet, where a rectangular symbol box is used.

3.4.1 Global Observations

Lighting Conditions

As can be seen in Figure 3.4, the lighting conditions in the images vary for all symbols. While some samples are bright, others are relatively dark, but still provide sufficient contrast for a human to recognize the symbol. This corresponds well to the intended application of the models in production, where children can take photos in arbitrary lighting conditions.


(a) Orientation Symbol Examples

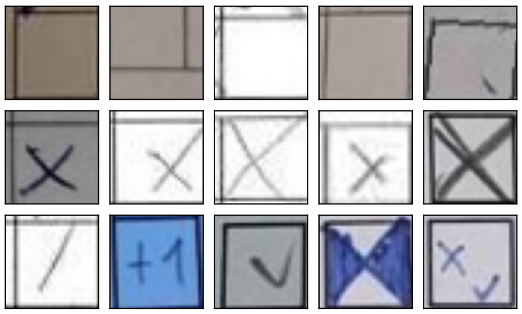

(b) Checkbox Symbol Examples

Figure 3.4: EXAMPLES OF (A) ORIENTATION AND (B) CHECKBOX SYMBOLS. The last row contains unknown symbols for the respective task type.

Handwriting Color and Thickness

Another aspect where the symbols differ is the writing color. This is visible, for example, in Figure 3.3, where the first -1 symbol is written in blue, while the other -1 symbols are written in black color. Further variation occurs in the thickness of the handwritten symbols, as can be seen in the same figure. The first -1 is written thick, while the second -1 is written rather thin.

Font Size

As the term bounding box suggests, symbols are written in different font sizes within the printed bounding box. Some children use the full box or even write slightly outside the bounding box, and others only use part of this box, which leads to slightly different locations of the symbols within the bounding box.

Bounding Box

Because the Herby localization introduces a slight error rate in the localization, the bounding box is part of the cropped symbols and appears in different relative locations. This is visible for the empty symbol row in both Figures 3.3 and 3.4. While its color is solid black, its contrast to both the background and the written symbol also varies with the aforementioned lighting conditions. The thickness and aspect ratio of the bounding box also vary due to the different data collection methodologies described in Section 3.2.

Unknown Symbols

A common type of unknown symbol is that children write symbols of another task type. For example, they write an instruction to move forward (+1) into a checkbox field as depicted in Figure 3.4. A further type of unknown symbol is when children change the symbol in an unclear manner, either by crossing it out as shown in Figure 3.3 or by simply writing the new symbol over the previous one as visible in Figure 3.4. Also, there are some incomplete symbols; for example, in Figure 3.3 there are rotation instructions that miss the tip of the rotation arrow and a move command (1) with no direction. A similar unknown symbol occurs for the checkbox task where only half a cross is written.

3.4.2 Specific Observations

Different Versions of the Digit One

Figure 3.3 shows the two variations of the digit one, which are present in the dataset. For both the +1 and -1 symbols, some children write the digit as a straight line only, while others include the edge at the top.

Turn Arrows

The same figure also shows the variety of both the arrows indicating a clockwise as well as an anti-clockwise turn. Some are written with an angle of more than 180 degrees, while others have a much lower angle. Another variety comes with the tip of the arrow, which is strongly emphasized by some children, while others draw only a small edge. Some children also fill the tip, while others draw it transparently.

Loop Symbols

The intended notation of the loop symbols is to draw the number plus the upper and left edges of the box as the start symbol of the loop and to draw the bottom and right edges as the end symbol of the loop. As can be seen in Figure 3.3, there is a large variation, as many children draw a full box, sometimes shadowing the respective edges to indicate the start, respectively, the end of the loop, and sometimes children even draw a fully filled-out box.

Unknown Symbols

The figures also reveal some specific types of unknown symbols that occur repeatedly. The most common one is reversing the order of conjunct instruction symbols as shown in Figure 3.3: Many unknown symbols occur where children write 1- or 1+ instead of -1 or +1. For the loop symbols, there are two sorts of unknown symbols that occur repeatedly as depicted in the same figure: The first is trying to execute a loop more than four times by writing a number greater than 4 before the loop start symbol. The second type is an instruction to move more than one field forward (or backward) by writing, for example, +8.

Chapter 4

Approach

This chapter explains the approaches used in the experiments during training and evaluation for both closed set and open set classification. It explains the novel approach to evaluating open set models for imbalanced datasets and is rounded off by listing the protocols and hyperparameters used in the experiments.

4.1 Models predicting Probabilities

The experiments use the two models described in Section 2.3.1. The LeNet-5 implementation from the PyTorch tutorials is used[1] which differs from the original implementation as described in the respective section. The output size of the models is adjusted to the respective number of output classes of each task type. The SimpleNet network is adapted based on the original author's PyTorch implementation.[2] The SimpleNet model is slimmed down by dividing the number of channels of all convolution layers by 4 in order to reduce model training time. In the closed set experiments, the model uses a single, fully connected layer after the global max pooling and random dropout with a probability of 0.1 to perform the classification. This model is referred to as *Slimmed SimpleNet* due to the reduced number of channels in the convolution layers. For the open set experiments, the slimmed SimpleNet is slightly adapted to what is referred to as *Slimmed SimpleNet EOS*. The adaptation is that the EOS version uses a linear layer with no bias to do the final classification.
The output of the models is transformed into probabilities. For orientation and instruction task types, which both have more than two known output classes, this is achieved by using PyTorch's softmax function.[3] For the binary checkbox classification task this probability is obtained using a sigmoid function $\sigma(x)$ on a single output neuron.[4] The respective formulas are listed in Equation 4.1. To obtain per-label probabilities in the binary case, the value of the sigmoid function $\sigma(x)$ is interpreted as the probability of label 1 and $1 - \sigma(x)$ as the probability of label 0. The highest model probability for a label is referred to as the *model confidence* throughout the thesis.

$$Softmax(x_i) = \frac{e_i^x}{\sum_j e_j^x}, \qquad \sigma(x) = \frac{1}{1 + e^{-x}} \tag{4.1}$$

[1] https://pytorch.org/tutorials/beginner/blitz/neural_networks_tutorial.html
[2] https://github.com/Coderx7/SimpleNet_Pytorch/blob/master/models/simplenet.py
[3] https://pytorch.org/docs/stable/generated/torch.nn.Softmax.html
[4] https://pytorch.org/docs/stable/generated/torch.nn.Sigmoid.html

Although the slimmed SimpleNet architecture is more complex than the LeNet-5 architecture, the number of trainable parameters is still manageable. For a model with 5 output classes, the proposed slimmed SimpleNet has 346'405 trainable parameters compared to the LeNet-5's 61'281 trainable parameters. In comparison, the original SimpleNet would have more than 5 million trainable parameters.

4.2 Data Augmentation

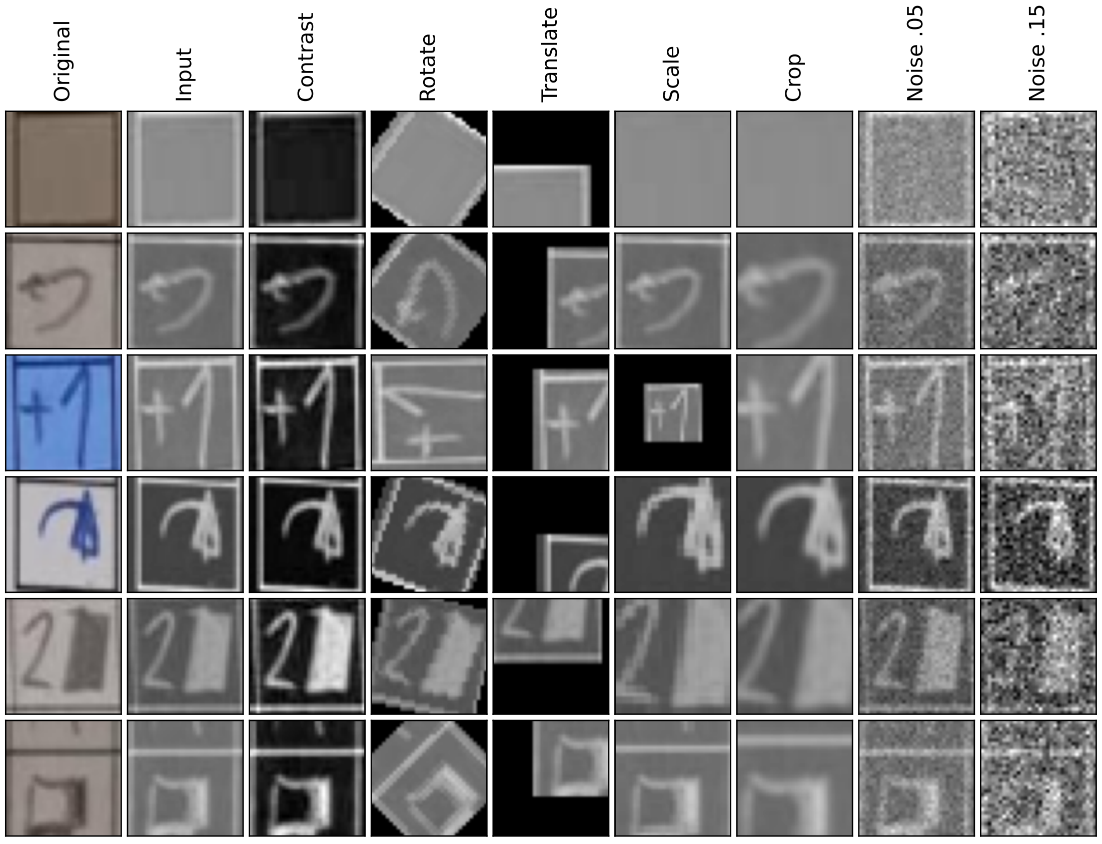


Figure 4.1: AUGMENTATION OF SELECTED SAMPLES. The first column shows the original symbol and the second column the input to the model, which is grayscaled and inverted. All other columns show a type of augmentation that is used (alone or in combination with others) in the experiments.

Data augmentation is used in the closed set classification to enhance the size and quality of the training data. In open set classification, a different data augmentation technique is used to generate unknown symbols to be used in training. There exist various different techniques, ranging from simple geometric transformations, such as rotations and cropping, to more complex solutions, such as training adversarial generative networks to artificially generate more samples Shorten and Khoshgoftaar (2019). Data augmentation in the experiments is limited to different geometric transformations and autocontrast implemented using PyTorch transforms.[5] Semantic restrictions apply to the transformations: A significantly too strong rotation of 90 degrees in the orientation task causes a sample to change its meaning, for example, from looking left ($<$) to looking up ($\wedge$). Analogously, cropping and scaling factors should be kept in a range where the whole symbol is still entirely visible, as otherwise the minus or plus in front of the one may be lost, obfuscating the meaning of the symbol.

Throughout the experiments, all symbols are grayscaled and rescaled to 32x32 pixels. Furthermore, the grayscale values are inverted as to have a higher input value for the drawn symbols instead of the background. Figure 4.1 shows the different transformations used in the experiments for six different symbols. The second column shows that grayscaling and inverting the symbols can already eliminate some of the lighting and color differences from the original photograph, but the contrast difference remains. Hence, the third column applies autocontrast[6] to the inverted symbol, leading to a solid black background and a solid white symbol and bounding box for different lighting conditions. Columns 4-7 show geometric transformations on the inverted grayscale symbol and its bounding box with the respective parameter chosen randomly from a uniform distribution. The fourth column shows rotations up to 90 degrees in both directions. A translation with a factor of 0.5 along both axes is shown in the fifth column and scaling with factors in the range [0.5, 1.5] is shown in the sixth column. While the center remains the same for rotating, scaling, and cropping the center of the bounding box, a translation along the x- or y-axis shifts the symbol along the respective direction. The seventh column shows a center crop of the symbol, which uses the original bounding box predicted by Herby.

The last two columns show the addition of Gaussian noise, which is the only data augmentation technique used to generate unknown symbols to be used in training in open set experiments. Gaussian noise is added to known samples after the pixels are already scaled to be in the range [0, 1]. Gaussian noise adds a random tensor that is sampled from a normal distribution with mean 0 and standard deviation 1 multiplied by σ to the original symbol. The first of the two Gaussian noise columns shows a σ of 0.05, and the second noise column shows a stronger perturbation value using a σ of 0.15.

In the experiments, the following data augmentation schemes are compared:

none No transformations, scaling is performed by dividing the grayscale pixel values by 255. This transform serves to provide a baseline for the use of no data augmentation scheme in training.

ac Autocontrast transformation, which is equivalent to scaling the symbols relative to their darkest and brightest pixel. Hence, the scaling is performed (for both training and test samples) using the following formula: $X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}}$ where X_{min} is the minimum pixel value in the symbol and X_{max} the maximum pixel value in the respective symbol. This augmentation (or scaling adaptation) serves to evaluate whether the different scaling alone can already improve classification performance.

geo Basic geometric transformations using the PyTorch RandomAffine transformation[7] that applies different geometric transformations in combination: It combines random scaling by a factor in range [0.85, 1.15] with random translation along the x-axis and y-axis using the range [-0.2, 0.2] and a random rotation in range [-20, 20] degrees. This augmentation scheme serves to evaluate whether the use of geometric transformations can improve classification performance.

[5]https://pytorch.org/vision/stable/transforms.html

[6]Linearly Rescaling pixels such that the minimum value is 0 and the maximum value is 255

[7]https://pytorch.org/vision/stable/generated/torchvision.transforms.RandomAffine.html#torchvision.transforms.RandomAffine

geo_ac First applies autocontrast and then the geometric transformations from geo. This augmentation scheme overcomes the limitation seen in Figure 4.1, where for geometric transformations, the background resulting from the geometric transformations is darker than the darkest pixel of the cropped symbol. It serves to evaluate whether a combination of geometric transformations with autocontrast can improve classification performance.

crop Applies a center crop that uses the original Herby bounding box.[8] This serves to evaluate whether the decision to crop larger than the originally predicted bounding box from Herby helps to improve classification performance.

crop_plus This augmentation scheme first applies the same center crop, followed by autocontrast and geometric (geo) transformations. It serves to evaluate whether the performance with the original bounding box can be improved by applying geometric transformations and autocontrast.

4.3 Evaluation Metrics

While a full analysis of model performance and model errors goes far beyond a single number or curve, this section explains the two evaluation metrics used in the experiments. Both evaluation metrics account for the imbalance in the test sets.

4.3.1 Balanced Accuracy

In the closed set classification, the model prediction corresponds to the label for which the model returns the highest probability. Every label can be classified as either the target label or one of the other labels. Therefore, for N labels, classifications can be visualized using a N x N confusion matrix, where the rows correspond to the target (or true) label and the columns to the predicted label. Figure 4.2 shows an example of such a confusion matrix for an imbalanced test set with three labels. The example serves to illustrate a weakness of using the accuracy for imbalanced classification as the accuracy divides the correct classifications (corresponding to all values in the diagonal of the confusion matrix) by the total number of samples. In the example, the accuracy would be relatively high with a value of $Accuracy = \frac{2+40+3}{50} = 90\%$. However, only 2 out of 5 samples of the label 0 are correctly classified. The recall obtained on this label corresponds to the number of correctly predicted samples of this label divided by the total number of samples of this label: $Recall_0 = 2/5 = 40\%$. This value is less than half the accuracy value. Balanced accuracy is calculated as the average recall obtained on each label, which corresponds to $Accuracy_{balanced} = \frac{Recall_0+Recall_1+Recall_2}{3} = \frac{0.4+1+0.6}{3} \approx 66.67\%$ for the confusion matrix shown in the example. Because performance on all the labels is important for a correction algorithm that builds on the model output, balanced accuracy is used as the main metric for the closed set classification, both as the validation metric as well as to report model performance on the test set.

4.3.2 Balanced Open Set Classification Rate Curve

An appropriate evaluation metric for the open set experiments has to consider that with entropic open set loss, the goal is to minimize model confidence for unknown samples. Further, it is desirable that it takes into account the label imbalance of the known samples in the test set. While the open set classification rate curve (OSCR) considers the aspect of the separation of known

[8]Slight deviations are to be expected because the symbol is first scaled to 32x32, then cropped to 22x22 and subsequently scaled to 32x32 again

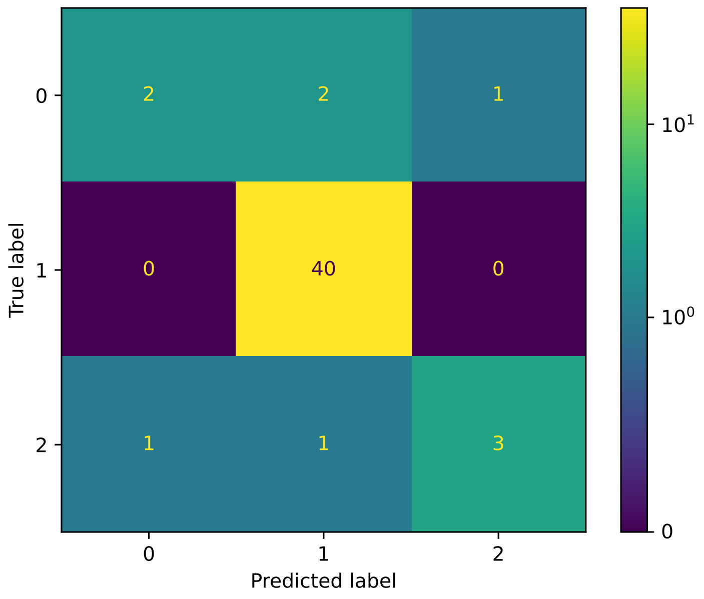


Figure 4.2: EXAMPLE OF A CONFUSION MATRIX ON AN IMBALANCED TEST SET. Rows are target labels, columns correspond to the actually predicted labels and each value represents the absolute number of samples the model predicts for this case.

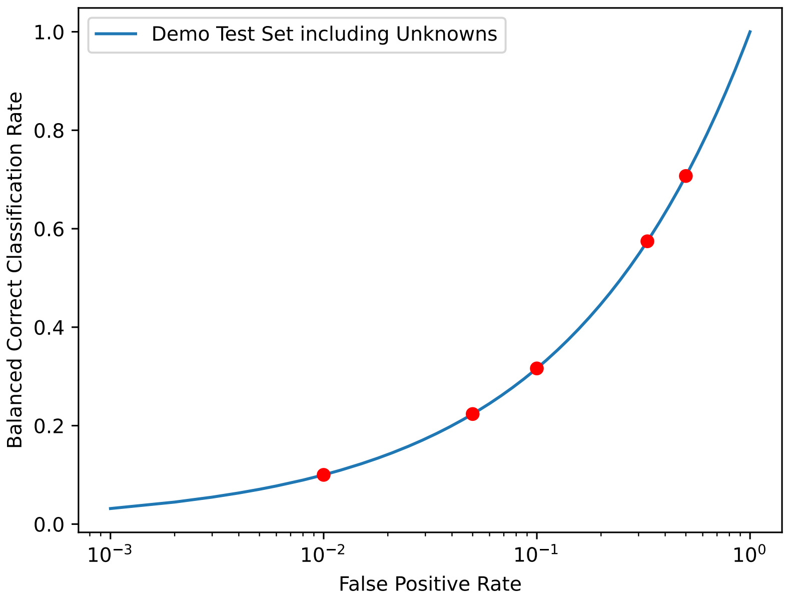


Figure 4.3: EXAMPLE OF A BALANCED OSCR CURVE. The Balanced Correct Classification Rates at selected FPRs of [0.01, 0.05, 0.1, 0.33, 0.5] are marked as red points.

and unknown samples based on the model confidence Dhamija et al. (2018), it does not take into account label imbalance of the known samples Anacona et al. (2022). To include the aspect of label imbalance as well, this thesis proposes an adaptation to the OSCR curve that takes into account label imbalance in known samples and can be called *Balanced Open Set Classification Rate Curve* (Balanced OSCR curve). To explain the adaptation made, the basic principle of the original OSCR curve has to be considered first. The OSCR curve iterates over different θ values and calculates the correct classification rate (CCR) and the false positive rate (FPR) for each of those values. The value θ serves as the decision boundary for the model confidence. It decides whether a sample is rejected as unknown or classified as the respective label with maximum probability. The sample is rejected as unknown if its model confidence P_{max} meets the criterion $P_{max} \leq \theta$. FPR is defined as the fraction of unknown samples in the test set that is classified as a known label out of all unknown test samples with respect to the decision boundary θ. CCR is defined as the fraction of known samples in the test set that is correctly classified with a model confidence of $P_{max}(x) > \theta$ out of all known samples in the test set. This is expressed as follows by the authors of the OSCR curve (incorporating the adaption proposed in Anacona et al. (2022)), where the test set is divided into two subsets D_l and D_u, where D_u represents the subset of unknown samples in the test set and D_l represents the subset of known samples in the test set. Each known sample has a known target label $\hat{c}$. $P(c|x)$ is the probability of a known label $c \in C$ for the sample x and C is the set of all known labels.

$$FPR(\theta) = \frac{\left|\{x | x \in D_u \wedge max_c P(c|x) > \theta\}\right|}{|D_u|} \tag{4.2}$$

$$CCR(\theta) = \frac{\left|\{x | x \in D_l \wedge arg\ max_c P(c|x) = \hat{c} \wedge P(\hat{c}|x) > \theta\}\right|}{|D_l|} \tag{4.3}$$

Similar to the difference between balanced accuracy and accuracy, the proposed adaptation is to calculate the CCR per known target label (label-wise) and then average it to obtain what can be called *Balanced Correct Classification Rate (BCCR)*. The balanced OSCR curve then uses the BCCR instead of the original CCR on the y-axis and the FPR on the x-axis to show the trade-off between achieving high classification performance on the known samples in the dataset and rejecting unknown samples by returning a low model confidence for them. The formula for calculating the CCR for a single label is the same as for the global CCR, but instead of using the subset of all known samples in the test set as D_l, the label-wise CCR only uses the subset of the test set which has the respective target label as D_l.
Figure 4.3 shows an example of a balanced OSCR curve using the function $sqrt(x)$. Throughout the experiments, a list of all the probabilities of unknown samples in the test set, as well as a value of zero, are used as the values of θ. Additionally, a list of FPRs ([0.01, 0.05, 0.1, 0.33, 0.5]) and the BCCR achieved for each rate are provided in tabular form in the appendix for the experiments considered. The values for those FPRs are obtained by performing linear interpolation on the values of the balanced OSCR curve.

4.4 Protocols

All experiments use the entire set of known labels per task type. By differentiating which type of unknown samples is used in training and validation (as known unknowns or known unknown classes, in short KUC) and which type of unknown samples is used in testing (as unknown unknowns or unknown unknown classes, in short UUC), the protocols can be divided into four types of protocols. The first type are closed set protocols, which build on the data splits described in Section 3.3 and do not use any KUC or UUC. The second type of protocol uses all unknown samples of the respective task type as UUC but does not use any KUC in training. The third type of protocol uses a fixed number of different symbols as KUC and all unknown samples of the respective task type as UUC. In the third protocol, KUC are created in three different ways. The first approach is to generate random pixel values using the torchvision FakeData dataset[9]. The second approach is to use randomly selected letters from the EMNIST dataset Cohen et al. (2017) and the last approach uses Gaussian noise with a σ of 0.05 on the respective amount of randomly selected training samples. Generally, all KUC samples use the same data augmentation scheme as the known labels in training. In the case of Gaussian noise, first, the Gaussian noise is applied and subsequently the augmentation scheme. The fourth type of protocol uses the S2 data split to assign unknown samples to KUC or UUC. This protocol can be referred to as a case of mixed unknowns, as some types of unknown samples may be present in both the KUC and the UUC. A particular example is that students from different school classes have drawn 1+ instead of +1 as an unknown instruction symbol.

Closed Set Closed Set Protocols compare the data splits S1, S2 and S3 described in Section 3.3 without using any KUC or UUC. The exact number of samples per label and data split is listed in the appendix in Section A.1.2.

Closed Set + UUC This protocol uses the S2 data split for the known samples. All unknown samples of the respective task type are used as UUC.

Open Set This protocol also uses the S2 data split for known samples. Additionally, 1’700 samples of KUC are added to the training and 300 such samples are added to the validation set. Again, all unknown samples of the respective task type are used as UUC.

Mixed Set This protocol not only uses the S2 data split for known samples, but also uses the S2 data split to differentiate between KUC and UUC. Therefore, the KUC and the UUC differ in the way that they originate from different school classes. As unknown samples are relatively scarce, additional random images are added to the KUC to achieve a total of 1’700 KUC samples in the training and 300 in the validation set.

4.5 Hyperparameters

Through all experiments, Adam is chosen as the optimization algorithm with a learning rate of 0.001 and betas of 0.9 and 0.999. Adam is a gradient-based optimization algorithm that computes individual adaptive learning rates for different parameters, and is designed to combine the advantages of the previously developed RMSProp and AdaGrad algorithms Kingma and Ba (2015). It is also used in related work Suter (2022), Dhamija et al. (2018).
Furthermore, in training, a batch size of 64 samples is used, and the training samples are randomly shuffled by the data loader in each epoch. Early stopping is used to stop the training process if the model does not improve on the validation metric for a given number of epochs.
The maximum number of training epochs is set to 125 for closed set models with early stopping allowing a maximum of 25 epochs without improvement in the validation metric. For the open set and mixed set experiments (including the closed set baseline model in those experiments),

[9]https://pytorch.org/vision/stable/generated/torchvision.datasets.FakeData.html

these numbers are increased to a maximum of 200 epochs and the training is stopped early after 50 epochs of no improvement on the validation metric.
The loss function used in closed set experiments is generally the cross entropy loss function[10] and its binary variant[11] is used for the checkbox task. Balanced accuracy is used as the validation metric. If there are unknown samples in the training set, the entropic open set loss is used both as the loss function in training and as the validation metric.
Entropic open set loss is equal to cross entropy loss for known samples. For unknown samples, it generates a target vector where the probability of each label is set to $\frac{1}{N}$ with N being the total number of known labels Dhamija et al. (2018).
The softmax baseline models in open and mixed set protocols, which do not use any unknown samples in training, use balanced accuracy as the validation metric, but all other hyperparameters are the same in order to maintain comparability. This baseline also uses the slimmed SimpleNet EOS adaptation, which does not have a bias in the classification layer.

[10] https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html
[11] https://pytorch.org/docs/stable/generated/torch.nn.BCELoss.html

Chapter 5

Experiments

This chapter contains the experiments, which use the protocols explained in Section 4.4. First, the general setup of the experiments is explained. Subsequently, different experiments are performed that provide a baseline for closed set and open set classification on the dataset.

5.1 Experiment Setup

All experiments are implemented using version 1.11 of the PyTorch library, which allows an easy implementation of DL models and supports the use of hardware accelerators during training Paszke et al. (2017). The implementation of the experiments, model checkpoints, and the anonymized dataset are all publicly available on Github.[1] A custom function first splits the data as described in the evaluation protocol and returns a data loader that shuffles the training samples randomly in each epoch. Subsequently, a CNN is trained on the training set and the parameters of the best models are selected based on the validation metric. Different visualizations of the training process, the model output and resulting classification errors are used to observe the training process and the results. Each experiment is structured in such a way that, first, the scope of the experiment is defined and the performance is reported using the evaluation metrics explained in Section 4.3. Subsequently, observations are made, and interpretations are given.

5.2 Closed Set Baseline

5.2.1 Scope and Results

The baseline experiment compares the two models described in Section 4.1 and the data augmentation schemes described in Section 4.2 for each data split and task type using the closed set protocol. The results for each of those combinations are shown in Figure 5.1. A numerical list of the balanced accuracy achieved by each model is provided in the appendix in Table A.4.

5.2.2 Difficulties of Data Splits and Task Types

By comparing the different data splits and task types for the same model and augmentation scheme, a trend becomes visible on which data splits and task types are harder than others. The

[1]https://github.com/aditen/hw-cs-sb-classifier

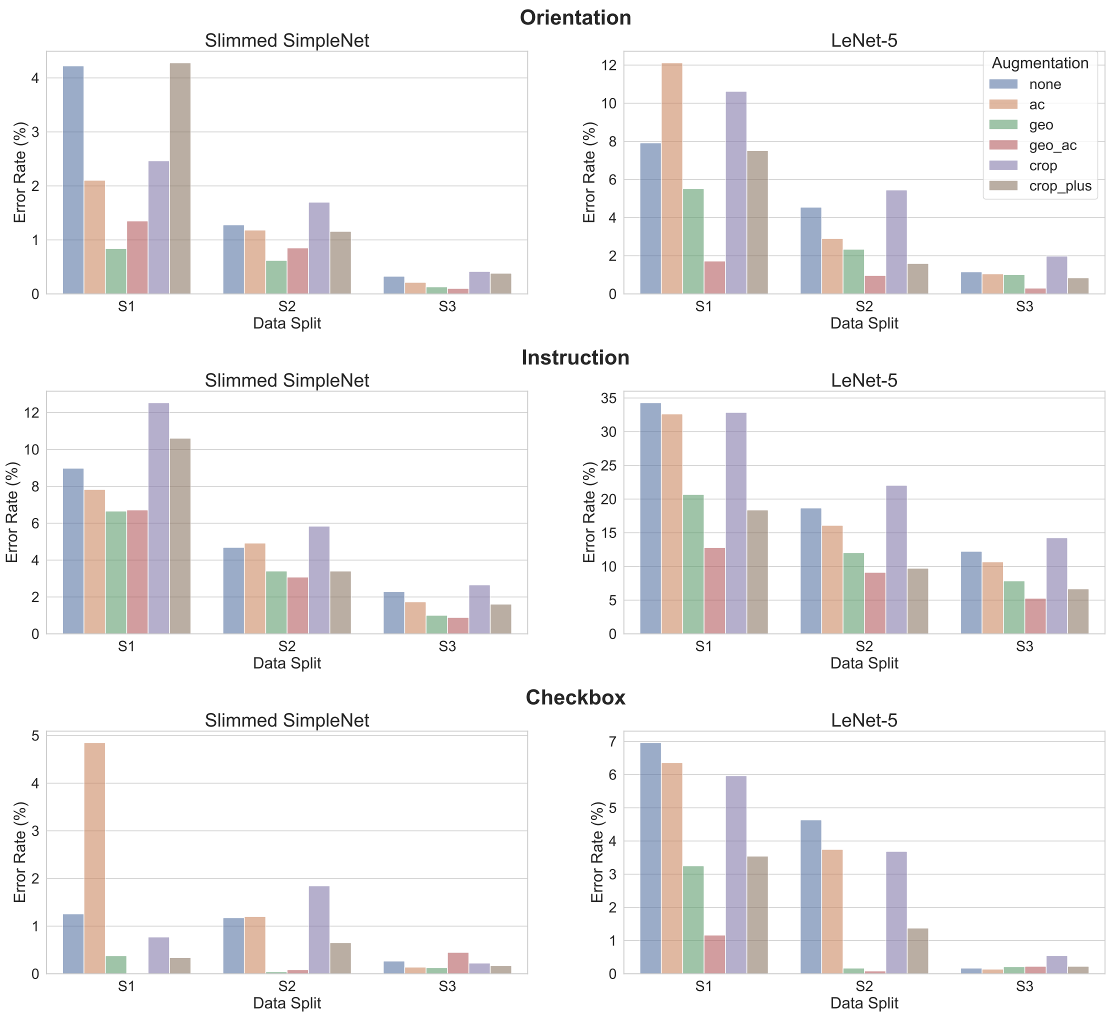


Figure 5.1: BASELINE ERROR RATE COMPARISON. Each task type represents a row, while in each row the performance of the slimmed SimpleNet is shown in the left and the performance of the LeNet-5 in the right subplot. Each of these subplots then shows the error rate for each data augmentation strategy on each data split. The error rate is obtained by subtracting the balanced accuracy from 100%.

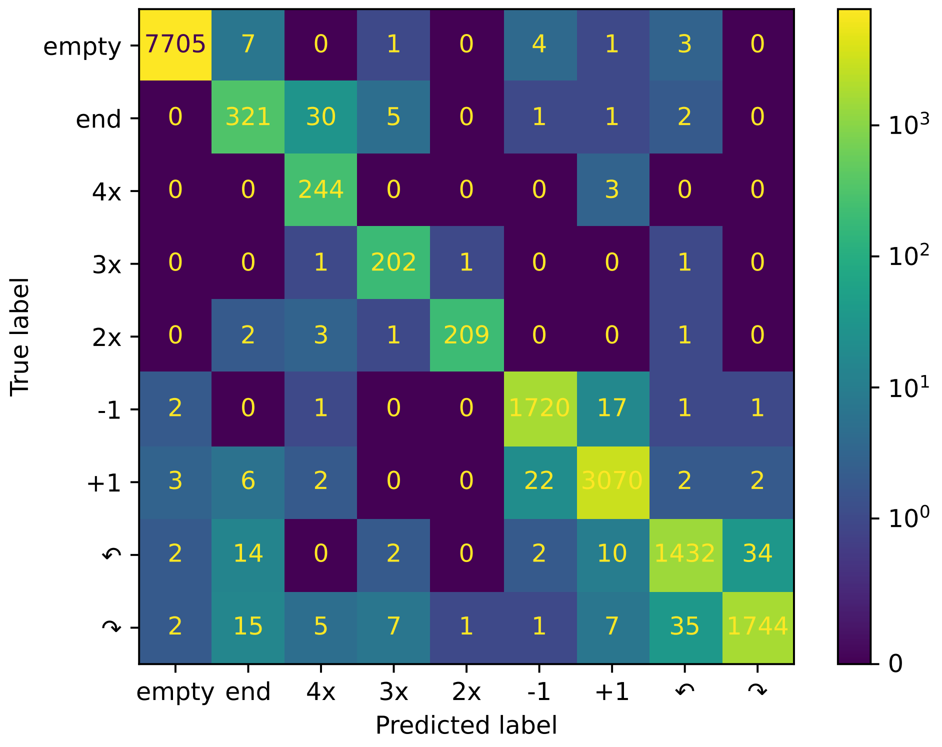
hardness is assessed by comparing the error rate: The higher the error rate achieved by the best model, the harder the task.
A general trend for the data splits is that the data split S3 is the simplest and S1 the hardest. A particular explanation for this phenomenon is the fact that in data split S3, some symbols of every child are already seen in the training set, and therefore the model can learn decision boundaries for that child. For both the data splits S1 and S2, the models must classify samples of new children in the test set. However, data split S1 contributes fewer training samples than S2, and those samples originate from a different type of exercise sheet, making this data split the hardest.
Another trend is visible in the difficulty of the task types: The instruction task type is the hardest, the orientation task type is the second hardest, and the checkbox task type is the easiest. Two different reasons can explain this phenomenon. The first reason is that the hardest type has the most labels and the easiest task type has the least labels, being a binary classification task. Second, the labeling guidelines permit minor variations in the instruction task type for the loop notation, which may increase the difficulty of this task type further.

5.2.3 Superiority of slimmed SimpleNet

The results show the superiority of the slimmed SimpleNet over the LeNet-5 in a number of scenarios, but not in all. This superiority can be explained by differences in model architectures. In particular, the slimmed SimpleNet is deeper and contains more convolution layers. Additionally, the included regularization may help the model to generalize better. However, this superiority mainly applies to the instruction task type, which is the hardest. For both the orientation and checkbox task types, the LeNet-5 model performs very closely to the SimpleNet for the best-performing augmentation scheme.

5.2.4 Influence of Data Augmentation

The influence of different data augmentation schemes on training data differs for the respective data splits and task types. For each data split and task type, at least a minor improvement over no augmentation can be found.
Different scaling introduced with the autocontrast augmentation can in some, but not all cases help to improve model performance. While there is only one case where the LeNet-5 performs worse with autocontrast than with default scaling, there are more such cases for the slimmed SimpleNet. The higher error rate on data split S1 when using autocontrast for the checkbox is the strongest, with the model performance degrading most.
Geometric transformations help for all models, task types, and data splits to reduce the error rate, except for the checkbox type with data split S3 and the LeNet-5 model. However, the absolute difference, in this case, is very small ($\leq$0.5%) and therefore should not be over-interpreted.
The crop augmentation, which uses the original bounding box predicted by Herby, performs worse than no augmentation in most scenarios. This can be explained by the fact that if the bounding box is predicted correctly by Herby, then the border of the bounding box is not included in the crop augmentation scheme, but it is included in all other schemes except crop_plus. However, if it is slightly off, then the border is included in both scenarios and therefore such localization offsets can add noise to the data in form of the border of the bounding box.
The combination of autocontrast and geometric transforms (geo_ac) helps to reduce the error rate the most in all cases except one for the LeNet-5. For the SimpleNet, this combination helps to reduce the error rate over no data augmentation in all cases but one. However, it is sometimes outperformed by geometric transformations with default scaling. Still, this augmentation scheme is considered the most promising and, therefore, is used in further experiments.



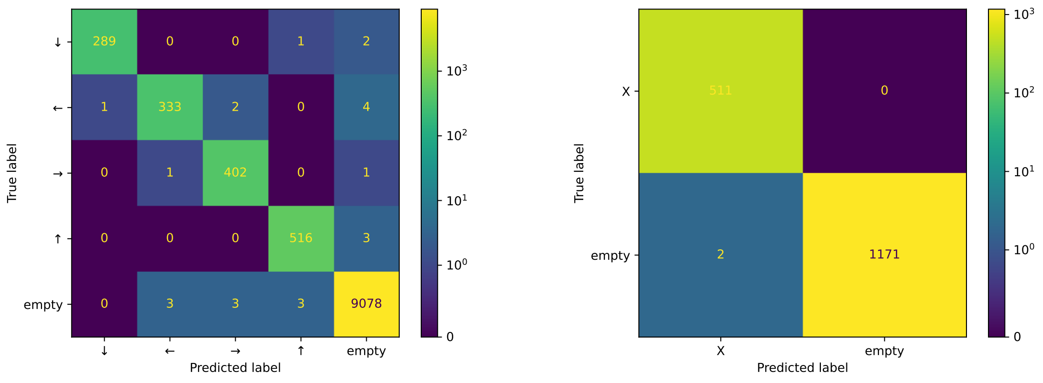
True label \ Predicted label	empty	end	4x	3x	2x	-1	+1	↶	↷
empty	7705	7	0	1	0	4	1	3	0
end	0	321	30	5	0	1	1	2	0
4x	0	0	244	0	0	0	3	0	0
3x	0	0	1	202	1	0	0	1	0
2x	0	2	3	1	209	0	0	1	0
-1	2	0	1	0	0	1720	17	1	1
+1	3	6	2	0	0	22	3070	2	2
↶	2	14	0	2	0	2	10	1432	34
↷	2	15	5	7	1	1	7	35	1744

Figure 5.2: Confusion Matrix of Instruction Model on Data Split S2. Every element represents the absolute number of samples in the test set for the respective classification case. Those are the numbers for the slimmed SimpleNet using the geo_ac augmentation scheme.

Applying the same geometric transformations and autocontrast to the bounding box originally predicted by Herby (crop_plus) also helps to improve performance over the crop augmentation scheme in most cases, which can be explained by the same reasoning as the improvement seen for geometric transforms over no augmentation. The phenomenon that this setup performs slightly worse than geo_ac, which crops the symbol with a margin of 25% added to the symbol on each side, can be explained by the possibility that when the predicted bounding box is slightly off, then geo_ac still gets to see a bigger part of the actual bounding box while crop may miss out on a semantically important part of the symbol like the arrow tip indicating the turning direction in the rotate instruction. This indicates that cropping larger than the predicted bounding box can help mitigate minor offsets in the preceding localization algorithm.

5.2.5 Model Errors

As model errors differ for each of the 108 trained models, model error observations are limited to the three slimmed SimpleNet models using the geo_ac augmentation scheme on data split S2 for each task type.
Figure 5.2 shows the confusion matrix of the slimmed SimpleNet on the symbols of the instruction task. Four specific types of error occur more frequently than others with respect to the total



(a) Orientation Symbols

(b) Checkbox Symbols

Figure 5.3: CONFUSION MATRICES OF (A) ORIENTATION AND (B) CHECKBOX MODEL ON DATA SPLIT S2. Those are the numbers for the slimmed SimpleNet using the geo_ac augmentation scheme. The confusion matrix also shows the strong label imbalance in test data for the orientation task, where more than 9'000 empty boxes are present and less than 2'000 symbols of the other labels combined.

number of samples of the respective target label. First, it happens that the direction of the instruction symbols is predicted wrongly by the model. A particular explanation for this phenomenon are differences in drawing the arrow tip and that children use different angles of rotation as the labeling guidelines allow any angle whose absolute value is greater than 10 degrees. The second common error is that +1 and -1 are confused. The +1 symbol can be seen as an extension of the -1 symbol with a vertical line across the minus. However, if this vertical line is very short or not drawn as thick as the minus, this may confuse the model. The third frequent type of error is that different symbols are misclassified as the end-of-loop symbol. This may be due to the model misinterpreting the bounding box as the end-of-loop symbol. Lastly, the end-of-loop symbol itself is frequently misclassified as the symbol to loop four times. An explanation for the fourth and also third types of error may be that the labeling guidelines are a bit more relaxed with regard to the loop notation than with regard to the other symbols.
The confusion matrices of the best orientation and checkbox models on data split S2 are shown in Figure 5.3. For both models, no type of error has a very high occurrence, as the balanced accuracy for both task types is $\geq 99\%$. In both task types, errors may arise because the bounding box is considered a part of the symbol by the model. For example, the conjunction of one edge of the orientation and a bounding box border may be misinterpreted as another orientation. Further, the borders of an empty bounding box may be misinterpreted as a symbol, or a symbol may be misinterpreted as the borders of an empty bounding box.

5.3 Closed Set Classifier on unknown Samples

5.3.1 Scope and Results

This experiment uses the slimmed SimpleNet models that were trained in the closed set baseline experiment on data split S2 using the geo_ac augmentation scheme. It follows the protocol closed set + UUC described in Section 4.4 and adds all the unknown samples of the respective task type

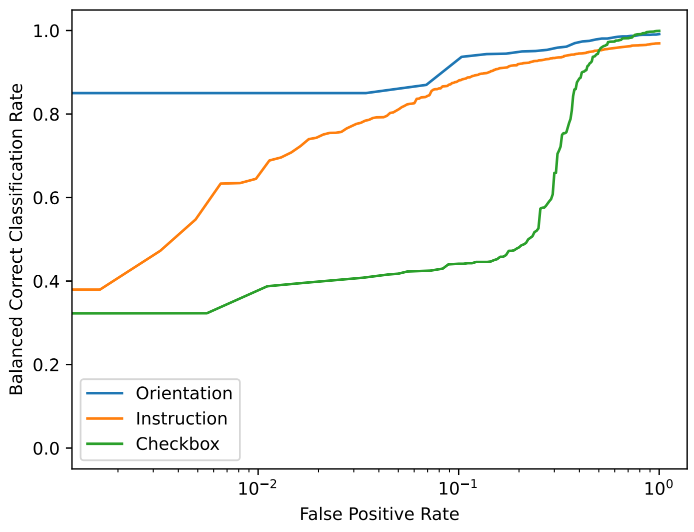


Figure 5.4: BALANCED OSCR CURVE USING S2 FOR KNOWN LABELS AND ALL UNKNOWN SAMPLES AS UUC. Each task type is represented as a separate curve.

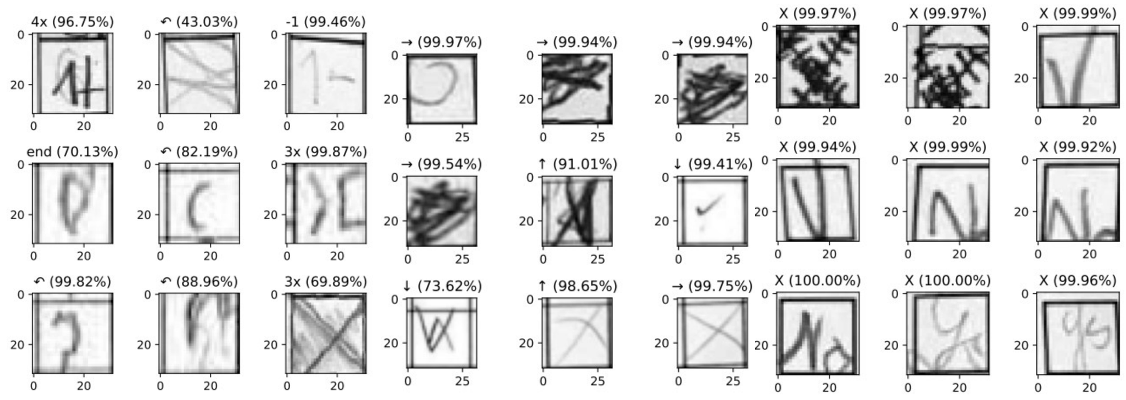
to the test set. The total number of unknown samples differs for each task type. There are only a total of 29 unknown samples for the orientation task type and 180 unknown samples for the checkbox task type. For the instruction task type, a total of 615 unknown samples are available. The balanced OSCR curve for each task type is shown in Figure 5.4. Selected false positive rates and their balanced correct classification rate are listed numerically in Table A.5 in the appendix.

5.3.2 Prediction and Model Confidence on unknown Samples

Looking at the model prediction for some of the unknown samples in Figure 5.5, it can be seen that the models of all task types produce relatively high model confidence for many unknown samples. While some of those can be explained by visual similarity like a 1- being classified as -1 with a high probability, other predictions are not that intuitive to explain. As an example, from two different crosses in an orientation symbol box, the first is classified as an arrow tip up with high model confidence, and the second is classified as an arrow tip right with high model confidence. An interesting phenomenon is that, for the checkbox task, all unknown symbols shown in the figure are classified as a checked box with a model confidence $\geq 99\%$. This may be explained by a checkbox task model that only learns to distinguish whether something is written into the box or not, but not whether the symbol written into this box actually corresponds to a cross or not.

5.3.3 Trade-Off Performance on known Samples vs Rejection of unknown Samples

The balanced OSCR curve in Figure 5.4 illustrates the trade-off between achieving high performance in the classification of known samples (corresponding to a high balanced correct classification rate) and achieving high performance in rejecting unknown samples (corresponding to a



(a) Predictions on unknown Instruction Samples

(b) Predictions on unknown Orientation Samples

(c) Predictions on unknown Checkbox Samples

Figure 5.5: PREDICTION ON (A) UNKNOWN INSTRUCTION, (B) UNKNOWN ORIENTATION AND (C) UNKNOWN CHECKBOX SAMPLES. Each subfigure contains 3x3 predictions on unknown samples of the respective task type. The title of each sample shows the predicted label and the model confidence.

low false positive rate). The trade-off when using the closed set models seems to be stronger for the checkbox and instruction task than for the orientation task. As there are only 29 unknown samples available for the orientation task, however, the meaningfulness of the OSCR curve of the instruction task model is limited, especially for low false positive rates.

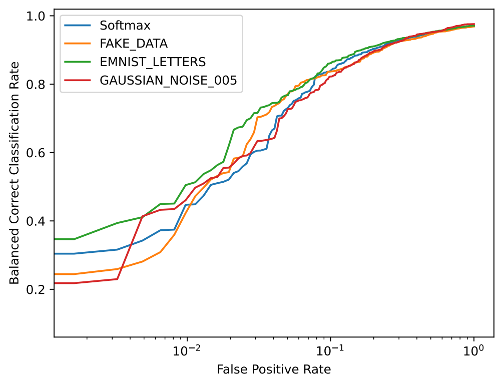
5.4 Training with known Unknowns

5.4.1 Scope and Results

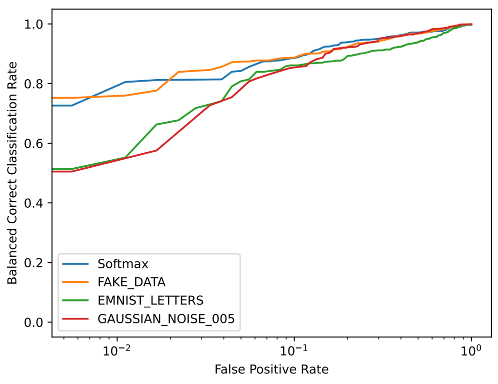
This experiment aims to evaluate whether using known unknowns in training helps to reduce the strength of the aforementioned trade-off in open set classification. It uses the open set protocol explained in Section 4.4, where a total of 2’000 known unknowns are added to the training and validation set and the models are trained using entropic open set loss. As hyperparameters are slightly different and this experiment uses the EOS version of the slimmed SimpleNet, a baseline using exactly the same model and hyperparameters (except the validation metric) is trained with no known unknowns in training. Models are evaluated on known samples from different school classes and unknown samples from all children as the test set, as in the previous experiment. Due to the limited amount of unknown samples for the orientation task (29), this experiment is limited to the instruction and checkbox task. The resulting balanced OSCR curves are shown in Figure 5.6. Selected points of the curve are listed numerically in Tables A.6 and A.7 in the appendix.

5.4.2 Impact on Trade-Off

The balanced OSCR curves show that for the instruction task type, all types of known unknowns are relatively close to the baseline. This indicates that for the instruction task, no significant improvement can be achieved by using those types of known unknowns during training. This can be explained by the known unknowns used for training being more distant from the known samples than the unknown samples in the test set. For example, both random images and letters



(a) Balanced OSCR curve for Instruction Task



(b) Balanced OSCR curve for Checkbox Task

Figure 5.6: BALANCED OSCR CURVE FOR USING KNOWN UNKNOWNS FOR (A) INSTRUCTION AND (B) CHECKBOX TASK TYPE. Both task types use a total of 2'000 unknowns in training (300 of them in the validation set).

do not include a bounding box around the symbol. While Gaussian noise uses noisy versions of known samples and therefore contains such a bounding box, it also cannot help to significantly improve this trade-off for the instruction task.
For the binary checkbox task, no visible improvement in the trade-off is found either. The same reasoning as for the instruction task may be used that the known unknowns seen in training are more different from the known symbols than the unknown symbols in the test set.

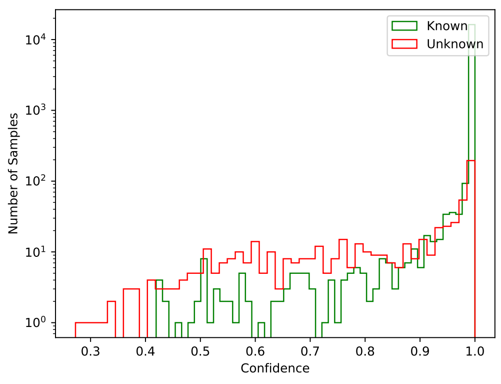
5.4.3 Difference in Model Confidence

To improve the trade-off illustrated by the balanced open set classification rate, the model has to produce reduced confidence for unknown samples and maintain high confidence for known samples. However, models still have high confidence for unknown samples in the instruction task, as illustrated in Figure 5.7. Most unknown samples maintain a confidence $\geq 95\%$ which is far away from the entropic open set optimization target to generate a probability of 11.11% for unknown symbols in the instruction task type.

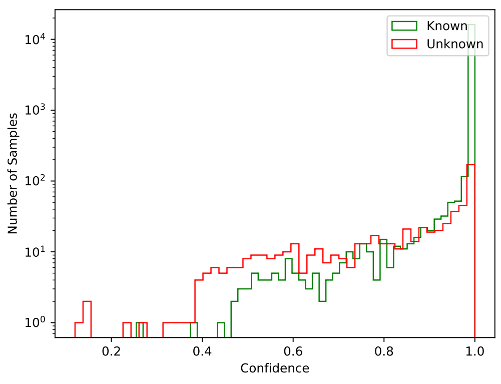
5.5 Mixed Unknowns

5.5.1 Scope and Results

To evaluate whether the addition of unknown samples in training data can help to reduce the strength of the open set trade-off, this experiment uses the mixed set protocol described in Section 4.4. Due to the scarcity of unknown samples for the orientation and checkbox task type, this experiment only covers the instruction task type. Therefore, unknown samples from only some of the school classes are used in training, contributing 296 unknown samples to the training set and 51 unknown samples to the validation set. 268 unknown samples from different school classes contribute to the test set. Hence, the results cannot be directly compared to the results of the previous open set experiment as the unknown symbols in the test set in this protocol are only a



(a) Confidence for no known Unknowns



(b) Confidence for using Letters as known Unknowns

Figure 5.7: Comparison of Confidence between (a) no known Unknowns and (b) Letters as known Unknowns on Instruction Task. The absolute number of samples at the respective confidence are plotted. The red bars represent unknown samples and the green bars represent correctly classified known samples from all known labels.

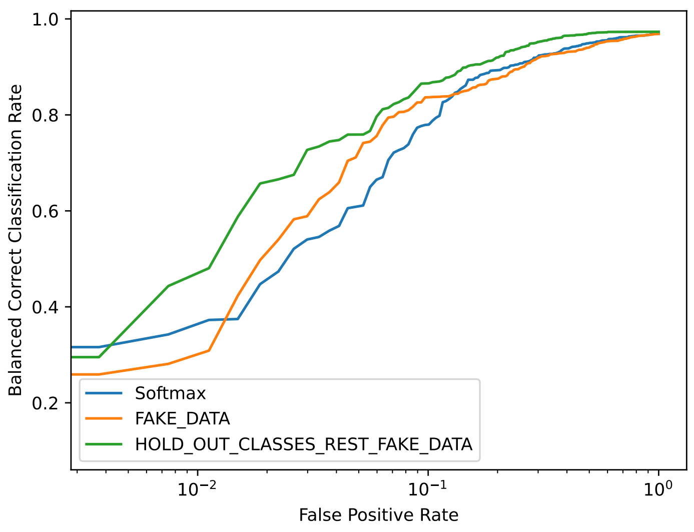
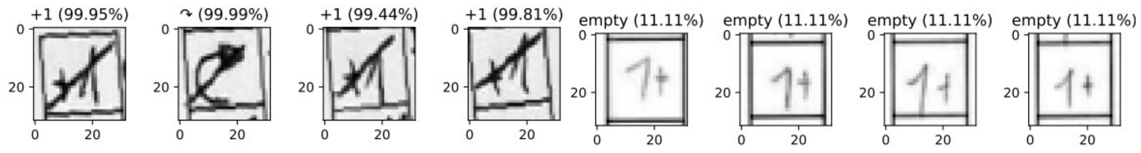
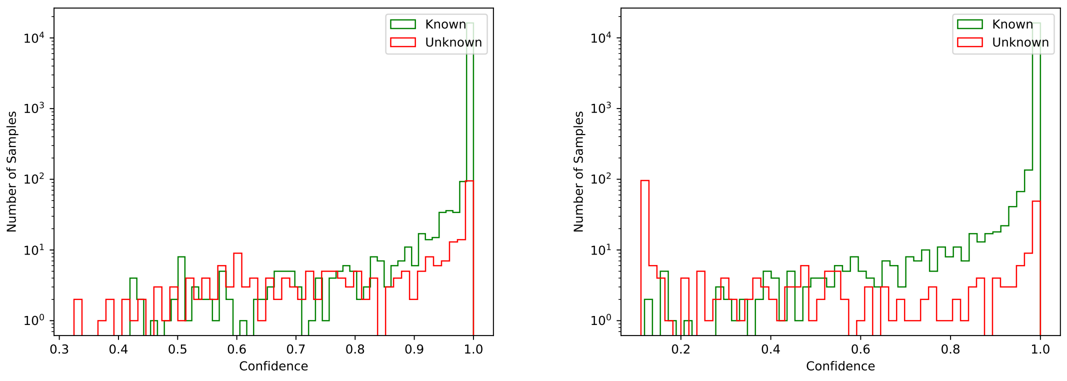


Figure 5.8: OSCR Curve using S2 to distinguish between known and unknown Unknowns. The blue curve represents the baseline that uses no known unknowns in training. The orange curve represents the use of only random images as known unknowns as a reference and the green curve represents the use of unknown samples from some students as known unknowns, which corresponds to mixed unknowns.



(a) Unknowns with high Model Confidence (b) Unknowns with low Model Confidence

Figure 5.9: Comparison of (a) Unknowns with high Model Confidence and (b) Unknowns with low Model Confidence in mixed Unknowns Protocol. Some types of unknown samples can be rejected with low model confidence while others maintain high model confidence.



(a) Confidence for no known Unknowns (b) Confidence for using known Unknowns of different Students

Figure 5.10: Comparison of Confidence between (a) no known Unknowns and (b) mixed Unknowns. Both subplots show the absolute number of samples at different confidence levels.

subset of the unknown symbols in the test set of the open set protocol. To have more unknown samples in training, the total number of unknowns in training is raised to 1'700 in the training set and 300 in the validation set by adding images with random pixel values. The resulting balanced OSCR curves are shown in Figure 5.8 and selected points of the curve are listed numerically in Table A.8 in the appendix.

5.5.2 Reduced Model Confidence on unknown Samples

The comparison of the model confidences for correctly predicted known samples and all unknown samples shown in Figure 5.10 illustrates that in the case of mixed unknowns, the model confidence for many unknown symbols can be drastically reduced compared to the baseline. However, a relevant number of unknown samples maintains high confidence. This can be explained by the different types of unknown symbols. The model learns to reduce the confidence for types of unknown symbols that are seen during training as known unknowns like a 1+ instead of +1. For the types of unknown symbols that occur only in the test set, on the other hand, the confidence remains high. This is shown in Figure 5.9, where the confidence for two different types of unknown symbols is compared.

Chapter 6

Discussion

This chapter discusses the collected dataset, the approaches used in the experiments, and the results of the experiments. This chapter further lists limitations and proposes future work that could be done.

6.1 Dataset

As part of the work, a dataset containing nearly 50'000 labeled samples was collected. To address *RQ1*, the information of school class affiliation present in the dataset is used to create a data split to emulate the productive use of the model (S2), which is to classify samples originating from new school classes. Another data split (S3) is created to provide an upper bound by randomly splitting the dataset. However, the S3 data split for this upper bound is very unrealistic to achieve in productive use, as it would require somebody to label correctly at least some samples of each child before it uses a model trained on these data to correct the exercises.
Another limitation of the dataset is that only little to no data are available from the productive exercise sheets (booklet) for some symbols. This limitation is shown to be not that severe because evaluating a specific data split S1 shows that a model trained on the data from the contributed sheet methodology achieves good performance on a test set from productive sheets and mini-booklets. An extended dataset from productive sheets may also provide further types of unknown samples and increase the meaningfulness of the experiment results.
Another important aspect of the dataset and the performed experiments are the chosen labeling guidelines, especially with regard to unknown symbols. Labeling samples differently would influence both the baseline performance and the open set trade-off between performance on known samples and rejecting unknown samples. Even having relatively clear labeling guidelines, differences occurred between the two annotators during the labeling process that had to be resolved by the master annotator.

6.2 Baseline

RQ2 is addressed by comparing a very simplistic model to a deeper model and comparing different combinations of geometric transformations and autocontrast on each data split in the baseline experiment. Both a deeper model as well as a particular combination of geometric transformations and autocontrast improve the classification performance over no data augmentation in most scenarios. Further performance improvements may be achieved by optimizing hyperparameters or by using a different model. A possible limitation is that, in the SimpleNet, there is only a dropout layer followed by a single linear layer after the final convolution block to perform the classification.

The baseline experiment also shows that by cropping the fields larger than the bounding box predicted by Herby, better performance can be achieved. However, this performance gap can be reduced in many cases by using geometric transformations and autocontrast during training.

Further, the input samples' original aspect ratio in the case of the instruction task is lost as the input symbols are all scaled to a square of size 32x32 throughout all task types and experiments in order to maintain comparability. This limitation also applies to the open set experiments performed in this thesis. Different strategies for dealing with rectangular symbol boxes could be evaluated in future work.

6.3 Open Set Classification

The open set experiments that address *RQ3* generally indicate that there is a relatively strong trade-off between achieving a low false positive ratio for unknown samples and a high classification performance for known samples. The separability of unknown and known samples based on the model confidence is only improved in the mixed unknown scenarios but still is limited to unknown samples that are very close to the known unknowns seen in training.

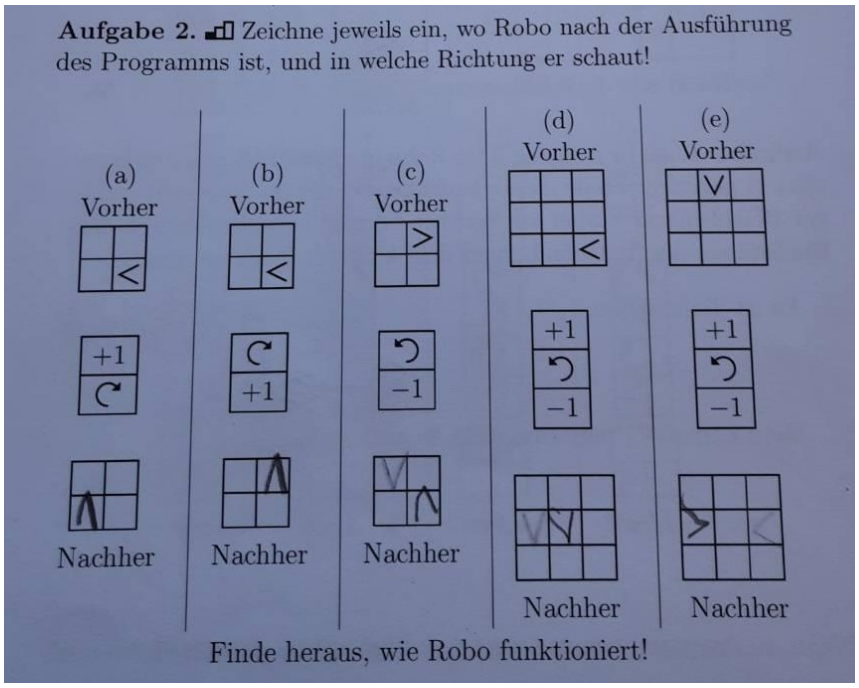
Further analysis could try to distinguish different types of unknown symbols, for example, by distinguishing them semantically: a first group could be crossed out symbols, a second group could be symbols of a different task type, and a third group could be misspelled symbols like 1+ instead of +1. Some of those types of unknown symbols could potentially be created synthetically like crossed-out symbols, which could be created by adding a line over a known symbol. Even better for the error type of children drawing the wrong symbol into the box, there would be hundreds to thousands of samples available for a different task type. As an example, an instruction model could be trained to reject orientation symbols as unknown and the data split S2 could be used to ensure that symbols from the same children do not appear in the training and test set.

Furthermore, the conducted open set experiments are limited to the entropic open set loss function. In further experiments, different open set approaches outlined in the background chapter could be compared.

6.4 Evaluation and Validation Metrics

The open set experiments use the balanced OSCR curve to plot the results. The adapted y-axis is regarded as an improvement over the traditional OSCR curve with respect to *RQ3* because the balanced correct classification rate is closer to the balanced accuracy, which is used as the closed set evaluation metric. However, the information visualized by this curve is still limited, similar to how the information when only providing balanced accuracy is limited. For example, the balanced correct classification rate does not show which type of error the model makes at a given point. To perform such an analysis, the confusion matrix would be more appropriate.

Also, the balanced OSCR curve only improves the analysis at test time. An appropriate validation metric for imbalanced validation data is not proposed in this thesis. However, a novel validation metric that averages two terms could be adapted in future work. Specifically, what the authors call *confidence metric* uses γ^+ to evaluate the classification of known samples, and γ^- to evaluate the rejection of unknown samples Anacona et al. (2022). In an adaptation for imbalanced datasets, the term for known samples (γ^+) could be calculated label-wise and then averaged in the sense of a balanced confidence metric, similar to the adaptation of the OSCR curve proposed in this thesis.


Aufgabe 2. Zeichne jeweils ein, wo Robo nach der Ausführung des Programms ist, und in welche Richtung er schaut!

Finde heraus, wie Robo funktioniert!

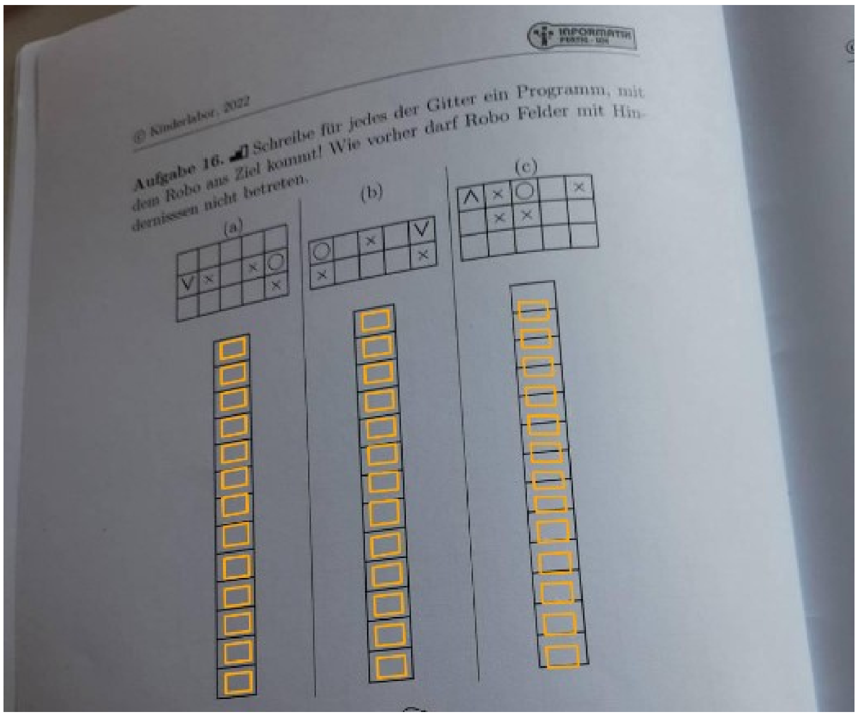
Figure 6.1: EXAMPLE OF MULTIPLE ORIENTATION SYMBOLS DRAWN ON A RESPONSE GRID. In tasks (c), (d) and (e) a child corrected the answer without fully erasing the old symbol.

6.5 Influence of Hyperparameters

The meaningfulness and interpretation of the results are generally limited to the design choices made in the experiments, especially with respect to the hyperparameters. Different results may be obtained on the same data when using different hyperparameters.
In addition, randomness also affects the experiment results as model parameters are initialized randomly and the data split S3 uses a given random seed. The impact of randomness could be assessed by repeatedly performing the experiments with different random seeds and then evaluating the distribution of values considering the mean and standard deviation of numeric results. A set of hyperparameters could be compared by performing a manual or grid search.

6.6 Independent Classification

A general limitation of this work is that the dataset and models are designed for independent classification. However, having the context of a whole exercise could help. As an example, it happens that children do not completely erase a symbol in the orientation task, leaving clearly visible drawing traces and drawing an even better visible robot in another field. This is shown in Figure 6.1. A human seeing the grid as context can clearly see that the location is at the corrected point, but a model doing independent classification can only decide based on a single field independently, hence may detect two robots on the grid and the predicted probability may not even be higher for the actual intended location.
Furthermore, the error rate expected in correcting entire exercises using independent classification is larger than the error rate of classifying an individual symbol. As an example, if six random



© Kinderlabor, 2022

Aufgabe 16. Schreibe für jedes der Gitter ein Programm, mit dem Robo ans Ziel kommt! Wie vorher darf Robo Felder mit Hindernissen nicht betreten.

Figure 6.2: EXAMPLE OF A LOCALIZATION ERROR. In task (c), the first seven boxes are all localized badly. Using this bounding box to crop the symbol, the classification model gets to see half of the actual box and half of another box.

symbols must all be classified correctly with such a model having a balanced accuracy of 95%, then the expected error rate ϵ may be estimated as $\hat{\epsilon}$ by subtracting the probability of a model predicting six random characters correctly: $\hat{\epsilon} = 1 - 0.95^6 \approx 26.49\%$. Instead of estimating this error, its true value could also be calculated directly by using the photograph ID and exercise information from the original dataset, which is both not published in this thesis for privacy reasons. Future work could also evaluate how a model could directly incorporate the exercise information.

6.7 Localization Errors

As per the labeling guidelines, severe localization errors are excluded from the dataset. However, such errors can occur in productive use as shown in Figure 6.2 with a photograph of an empty sheet where some of the boxes are localized in between two printed boxes. In such cases, cropping according to this predicted bounding box and then classifying the cropped part of the photograph independently is very prone to errors. Hence, an analysis of the localization errors as well as potential improvements by adapting the exercise sheets could be studied. A potential improvement could be to add more (local) orientation points to the exercise sheets that can be used by the localization algorithm. An important limitation of such adaptations is that they are not backwards compatible, hence the localization error of existing sheets cannot be reduced this way.

Chapter 7

Conclusion

In this thesis, an image classification dataset for a novel domain of handwritten symbols is created and different classification models are evaluated.
A particular challenge of the data from productive use is the scarcity of some symbols. To overcome this limitation, a custom exercise sheet is contributed as an alternative data collection methodology, where children are instructed to draw the same symbols in a different exercise context. The experiments show that the data from the contributed exercise sheet do not differ severely from the data from the booklet that is used in production, as a model trained exclusively on the data from this exercise sheet performs reasonably well in classifying the symbols of the productive booklet. Should more productive data originating from booklets become available in the future, this would allow to further assess the impact of the data originating from the contributed exercise sheet. As an example, two different data splits could be compared, where the first data split uses only the data originating from the booklets for training, and the second data split uses the data originating from both the booklets and the contributed exercise sheet for training.
The experiments evaluate (closed set) baseline, open set and mixed set classification models. Baseline models that have to classify handwritten symbols from new school classes achieve good performance for each task type. As the best error rate for each task type is less than 4% for samples from new school classes, this is considered sufficient to use the models in production for the classification step of the correction process. With additional labeled data, its performance could be further improved. A limitation of the error analysis is that it cannot use the information of which child drew the symbol, its school class affiliation and the whole photograph. If legal concerns are resolved, further analysis could be made using this information.
The open set experiments indicate that the task of rejecting unknown symbols is relatively hard because the unknown symbols are close to the known symbols. In the case of mixed unknowns, the model learns to at least reject unknown symbols that are similar to the unknown symbols seen in training. In order to elaborate on the usability of an open set model in production, the impact of a high false positive rate on the whole correction process has to be assessed, which is not considered in this thesis. A relatively high false positive rate of 33% or even more could already be sufficient considering the relatively low occurrence of unknown symbols in the dataset. For such false positive rates, the models can achieve relatively high balanced correct classification rates $\geq 90\%$.
Regarding the adaptation of the OSCR curve to a balanced OSCR curve, an improvement in the interpretability of the y-axis of the curve for imbalanced test sets is achieved. Still, other limitations of the OSCR curve cannot be overcome. Its suitability for imbalanced datasets could be further assessed by using different datasets and by considering specific edge cases. A validation metric considering both label imbalance and the rejection of unknown samples is desirable for the training process and a proposal is made in the discussion.
Regarding the target labels, the labeling guidelines are determined from the authors' point of

view. From a purely educational point of view, those could be set differently, treating both +1 and 1+ as the +1 symbol, for example. Different labeling guidelines would lead to different experiment results, based on which the usability of the approaches would have to be reassessed.
Summing up with respect to the productive use, this thesis provides baseline models which can be used in an initial version of a productive correction system for the step of independently classifying the handwritten symbols. The models can also learn to reject the types of unknown symbols that are seen in training with low model confidence, but rejecting all types of unknown symbols is hard to achieve without suffering from heavily reduced classification performance on known symbols. Further analysis could consider the correction system as a whole, including an educational point of view and a greater amount of productive data when available.

Appendix A

Attachments

A.1 Dataset

A.1.1 Labeling Guidelines

1. Symbols shall be labeled independently and context shall be ignored. Only what is written inside the printed bounding box and 10% above it shall be considered when labeling.

2. In case of a very bad Herby localization (<60% of the bounding box detected) the field shall be labeled for inspection unless all the fields within the detected box are empty, then it shall be labeled as empty.

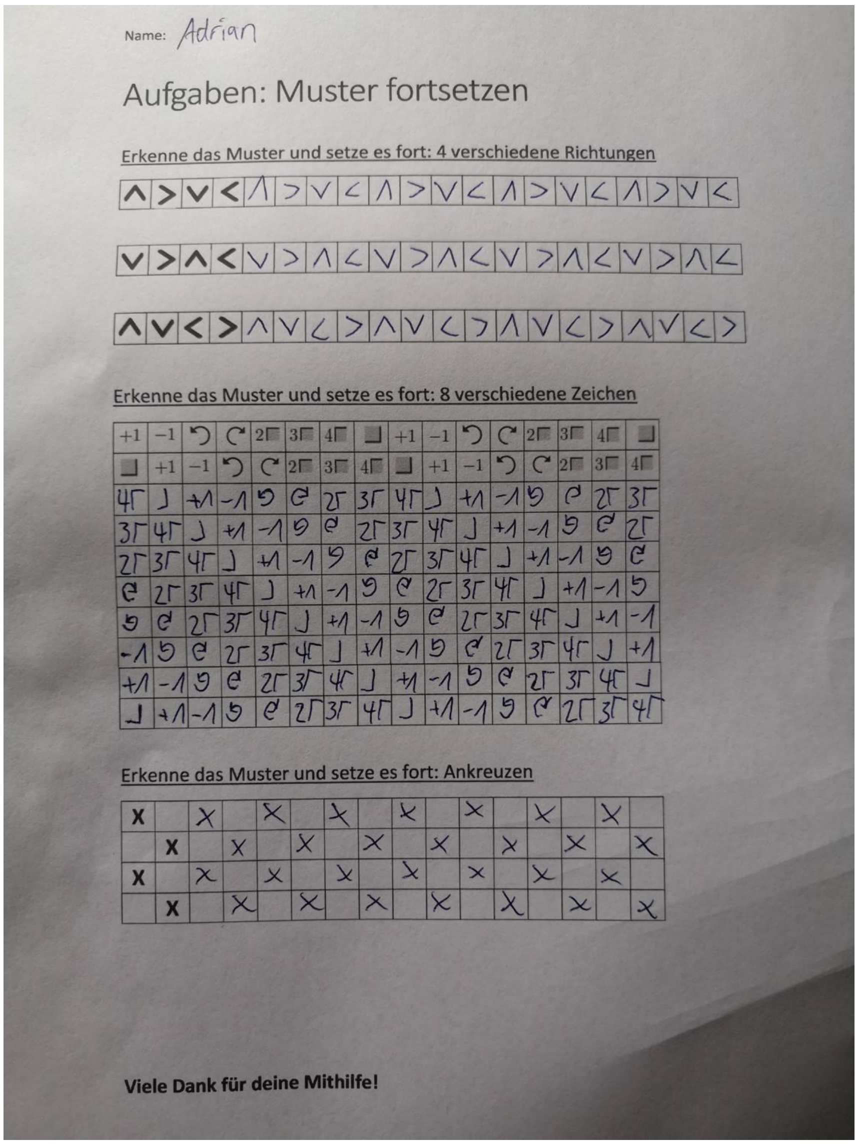
3. Symbols significantly lacking contrast, potentially because they were erased, shall also be labeled for inspection. Significantly lacking contrast means that the opacity of the symbol is less than 33% the opacity of the bounding box.[1]

4. If there are clear drawing traces (opacity $\geq 33\%$ compared to the bounding box) in the bounding box, but the symbol is not clearly assignable, it shall be labeled as unknown. Examples include A) wrong order of conjunct symbols (like 1+ instead of +1), B) mixing up completely the two edges of start and end-of-loop symbol (at least one edge has to be correct or a whole box has to be drawn), C) rotation instructions that are drawn with a rotation of maximum 10 degrees (like a straight arrow to the right or left) or if no tip at all is visible, D) orientation symbols where the angle is very close (+/- 10 degrees) to +/- 45 or +/- 135 degrees

A.1.2 Labels per Data Split

Tables A.1, A.2 and A.3 show the absolute number of samples for each symbol in the training, validation and test set. Values are separated using /, where the first number represents the number of training samples, the second number represents the number of validation samples and the last number represents the number of test samples. When referring to *all unknown samples* in the thesis, the numbers from the training, validation and test set in the question mark column can be added up to get the respective number.

[1] Can be visualized using CSS, an example is in the Mozilla MDN Docs: https://developer.mozilla.org/en-US/docs/Web/CSS/opacity?retiredLocale=de



Name: Adrian

Aufgaben: Muster fortsetzen

Erkenne das Muster und setze es fort: 4 verschiedene Richtungen

∧	>	∨	<	∧	>	∨	<	∧	>	∨	<	∧	>	∨	<	∧	>	∨	<

∨	>	∧	<	∨	>	∧	<	∨	>	∧	<	∨	>	∧	<	∨	>	∧	<

∧	∨	<	>	∧	∨	<	>	∧	∨	<	>	∧	∨	<	>	∧	∨	<	>

Erkenne das Muster und setze es fort: 8 verschiedene Zeichen

+1	−1	↺	↻	2Γ	3Γ	4Γ	■	+1	−1	↺	↻	2Γ	3Γ	4Γ	■
■	+1	−1	↺	↻	2Γ	3Γ	4Γ	■	+1	−1	↺	↻	2Γ	3Γ	4Γ
4Γ	■	+1	−1	↺	↻	2Γ	3Γ	4Γ	■	+1	−1	↺	↻	2Γ	3Γ
3Γ	4Γ	■	+1	−1	↺	↻	2Γ	3Γ	4Γ	■	+1	−1	↺	↻	2Γ
2Γ	3Γ	4Γ	■	+1	−1	↺	↻	2Γ	3Γ	4Γ	■	+1	−1	↺	↻
↻	2Γ	3Γ	4Γ	■	+1	−1	↺	↻	2Γ	3Γ	4Γ	■	+1	−1	↺
↺	↻	2Γ	3Γ	4Γ	■	+1	−1	↺	↻	2Γ	3Γ	4Γ	■	+1	−1
−1	↺	↻	2Γ	3Γ	4Γ	■	+1	−1	↺	↻	2Γ	3Γ	4Γ	■	+1
+1	−1	↺	↻	2Γ	3Γ	4Γ	■	+1	−1	↺	↻	2Γ	3Γ	4Γ	■
■	+1	−1	↺	↻	2Γ	3Γ	4Γ	■	+1	−1	↺	↻	2Γ	3Γ	4Γ

Erkenne das Muster und setze es fort: Ankreuzen

X		X		X		X		X		X		X		X	
	X		X		X		X		X		X		X		X
X		X		X		X		X		X		X		X	
	X		X		X		X		X		X		X		X

Viele Dank für deine Mithilfe!

Figure A.1: Example of the contributed Exercise Sheet. The task is to continue the pattern using the symbols printed into the boxes. The sheet was filled out as a test by the author.

Data Split	empty	↓	←	↑	→	?
S1	177/48/12663	621/108/213	621/90/290	626/110/548	618/114/372	1/0/28
S2	3231/570/9087	539/111/292	558/103/340	652/113/519	612/88/404	9/3/17
S3	1281/219/11388	642/101/199	671/124/206	885/164/235	735/134/235	19/5/5

Table A.1: Number of Samples for each Data Split in Orientation Task. The first number denotes the number of training samples, the second number is the number of validation samples and the last number is the number of test samples.

Data Split	-1	↺	empty	↻	+1	3x	end	4x	2x	?
S1	655/117/2179	650/125/1839	154/39/8384	661/104/2139	672/112/3861	668/121/66	746/110/293	666/117/152	658/130/86	93/17/505
S2	1030/179/1742	933/185/1496	732/124/7721	928/159/1817	1301/237/3107	551/99/205	677/112/360	584/104/247	567/91/216	296/51/268
S3	1289/211/1451	1285/215/1114	1269/231/7077	1283/217/1404	1279/221/3145	572/108/175	783/133/233	628/127/180	586/113/175	421/82/112

Table A.2: NUMBER OF SAMPLES FOR EACH DATA SPLIT IN INSTRUCTION TASK. The first number denotes the number of training samples, the second number is the number of validation samples and the last number is the number of test samples.

Data Split	empty	X	?
S1	1408/237/1030	1313/244/264	2/0/178
S2	1276/226/1173	1112/198/511	26/2/152
S3	1274/226/1175	1242/220/359	113/18/49

Table A.3: NUMBER OF SAMPLES FOR EACH DATA SPLIT IN CHECKBOX TASK. The first number denotes the number of training samples, the second number is the number of validation samples and the last number is the number of test samples.

A.2 Experiments

For the baseline experiment from Section 5.2, Table A.4 shows the balanced accuracy rounded to two decimals of all 108 baseline models.
Table A.5 shows the balanced correct classification rate for selected false positive rates when using some of those models to detect unknowns using a confidence threshold as described in Section 5.3.
Tables A.6 and A.7 show the balanced correct classification rate for the same false positive rates for the experiments of Section 5.4 using different types of known unknowns in training.
Finally, Table A.8 shows selected points in the OSCR curve for the experiment from Section 5.5, comparing the use of unknown symbols from different children during training.

Data Split	Augmentation	Instruction		Orientation		Checkbox	
		SimpleNet	LeNet	SimpleNet	LeNet	SimpleNet	LeNet
S1	none	91.02%	65.71%	95.78%	92.08%	98.74%	93.04%
	ac	92.17%	67.36%	97.90%	87.88%	95.15%	93.64%
	geo	93.34%	79.32%	99.16%	94.48%	99.62%	96.75%
	geo_ac	93.28%	87.19%	98.65%	98.28%	100.00%	98.83%
	crop	87.47%	67.14%	97.54%	89.38%	99.23%	94.03%
	crop_plus	89.39%	81.61%	95.72%	92.49%	99.66%	96.46%
S2	none	95.31%	81.32%	98.72%	95.45%	98.82%	95.36%
	ac	95.08%	83.90%	98.82%	97.10%	98.80%	96.25%
	geo	96.59%	87.96%	99.38%	97.66%	99.96%	99.83%
	geo_ac	96.92%	90.88%	99.15%	99.03%	99.91%	99.91%
	crop	94.16%	77.96%	98.30%	94.55%	98.15%	96.31%
	crop_plus	96.59%	90.27%	98.84%	98.41%	99.35%	98.62%
S3	none	97.71%	87.76%	99.67%	98.84%	99.73%	99.83%
	ac	98.26%	89.32%	99.79%	98.95%	99.86%	99.86%
	geo	99.00%	92.14%	99.87%	98.99%	99.87%	99.79%
	geo_ac	99.11%	94.73%	99.90%	99.70%	99.55%	99.78%
	crop	97.34%	85.74%	99.58%	98.02%	99.78%	99.45%
	crop_plus	98.39%	93.32%	99.62%	99.16%	99.83%	99.78%

Table A.4: BASELINE PERFORMANCES PER DATA SPLIT, AUGMENTATION SCHEME AND TASK TYPE. Each cell reports the balanced accuracy of the respective model. SimpleNet refers to the slimmed SimpleNet.

Task Type	0.01	0.05	0.1	0.33	0.5
Orientation	0.8413	0.859	0.9302	0.9603	0.9796
Instruction	0.6514	0.8108	0.8799	0.9368	0.9527
Checkbox	0.3746	0.4176	0.4414	0.7518	0.9508

Table A.5: BCCR@FPR FOR CLOSED SET CLASSIFIER PREDICTING ON UNKNOWN SAMPLES. Each column represents a False Positive Rate (FPR) and the entry the Balanced Correct Classification Rate (BCCR) achieved by the model.

Known Unknowns	0.01	0.05	0.1	0.33	0.5
None	0.7896	0.8425	0.8857	0.9552	0.9711
Random Images	0.7583	0.8737	0.887	0.9478	0.9683
Letters	0.5443	0.8078	0.8611	0.9143	0.94
Gaussian Noise	0.5333	0.7809	0.8539	0.9534	0.9681

Table A.6: BCCR@FPR FOR CHECKBOX OPEN SET CLASSIFIER USING DIFFERENT KNOWN UNKNOWNS. Values represent the BCCR for a given FPR per column. Gaussian noise uses a σ of 0.05.

Known Unknowns	0.01	0.05	0.1	0.33	0.5
None	0.4475	0.7297	0.8393	0.9301	0.9496
Random Images	0.4305	0.7702	0.837	0.928	0.9462
Letters	0.506	0.7675	0.8614	0.9353	0.9519
Gaussian Noise	0.4661	0.724	0.8224	0.9326	0.9517

Table A.7: BCCR@FPR FOR INSTRUCTION OPEN SET CLASSIFIER USING DIFFERENT KNOWN UNKNOWNS. Values represent the BCCR for a given FPR per column. Gaussian noise uses a σ of 0.05.

Known Unknowns	0.01	0.05	0.1	0.33	0.5
None	0.363	0.6097	0.7797	0.9261	0.9497
Random Images	0.3	0.7234	0.8367	0.9235	0.9404
Unknown Symbols + Random Images	0.4687	0.759	0.8656	0.9559	0.9699

Table A.8: BCCR@FPR FOR INSTRUCTION OPEN SET CLASSIFIER IN THE MIXED UNKNOWNS PROTOCOL. Values represent the BCCR for a given FPR per column. Gaussian noise uses a σ of 0.05.

List of Figures

List of Tables

Bibliography

Albawi, S., Mohammed, T. A., and Al-Zawi, S. (2017). Understanding of a convolutional neural network. In *2017 international conference on engineering and technology (ICET)*, pages 1–6. Ieee.

Anacona, J. A. P., Bhoumik, A., and Günther, M. (2022). Large-scale open-set classification protocols for imagenet. *arXiv preprint arXiv:2210.06789*.

Calvo-Zaragoza, J. and Oncina, J. (2014). Recognition of pen-based music notation: the homus dataset. In *2014 22nd International Conference on Pattern Recognition*, pages 3038–3043. IEEE.

Cireşan, D. and Meier, U. (2015). Multi-column deep neural networks for offline handwritten chinese character classification. In *2015 international joint conference on neural networks (IJCNN)*, pages 1–6. IEEE.

Cohen, G., Afshar, S., Tapson, J., and Van Schaik, A. (2017). EMNIST: Extending MNIST to handwritten letters. In *2017 international joint conference on neural networks (IJCNN)*, pages 2921–2926. IEEE.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Deng, L. (2012). The MNIST database of handwritten digit images for machine learning research [best of the web]. *IEEE signal processing magazine*, 29(6):141–142.

Dhamija, A. R., Günther, M., and Boult, T. (2018). Reducing network agnostophobia. *Advances in Neural Information Processing Systems*, 31.

Geng, C., Huang, S.-j., and Chen, S. (2020). Recent advances in open set recognition: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 43(10):3614–3631.

Gärtner, B., Adamaszek, P., and Klasovita, I. (2021). Informatik, fertig, los – Materialien für die 3. und 4. Klasse aus dem Kinderlabor. https://kinderlabor.ch/informatik-fertig-los-materialien-fuer-die-3-und-4-klasse/. [Online; accessed 02-June-2022].

Hasanpour, S. H., Rouhani, M., Fayyaz, M., and Sabokrou, M. (2016). Lets keep it simple, using simple architectures to outperform deeper and more complex architectures. *arXiv preprint arXiv:1608.06037*.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436–444.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lee, S., Son, S. J., Oh, J., and Kwak, N. (2016). Handwritten music symbol classification using deep convolutional neural networks. In *2016 International Conference on Information Science and Security (ICISS)*, pages 1–5. IEEE.

Liu, C.-L., Yin, F., Wang, D.-H., and Wang, Q.-F. (2013). Online and offline handwritten chinese character recognition: benchmarking on new databases. *Pattern Recognition*, 46(1):155–162.

Miller, D., Sunderhauf, N., Milford, M., and Dayoub, F. (2021). Class anchor clustering: A loss for distance-based open set recognition. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3570–3578.

Mouchère, H., Viard-Gaudin, C., Zanibbi, R., and Garain, U. (2016). ICFHR2016 CROHME: Competition on recognition of online handwritten mathematical expressions. In *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 607–612. IEEE.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. (2011). Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in pytorch. In *NIPS 2017 Workshop on Autodiff*.

Pelican, A. (2019). Herby. https://www.herby.digital/. [Online; accessed 02-June-2022].

Ramadhan, I., Purnama, B., and Al Faraby, S. (2016). Convolutional neural networks applied to handwritten mathematical symbols classification. In *2016 4th international conference on information and communication technology (ICoICT)*, pages 1–4. IEEE.

Shorten, C. and Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48.

Sonawane, P. K. and Shelke, S. (2018). Handwritten devanagari character classification using deep learning. In *2018 International Conference on Information, Communication, Engineering and Technology (ICICET)*, pages 1–4. IEEE.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Suter, M. (2022). Development and comparison of open set classification techniques on imagenet. Master's thesis, University of Zurich.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Tafti, A. P., Baghaie, A., Assefi, M., Arabnia, H. R., Yu, Z., and Peissig, P. (2016). OCR as a service: an experimental evaluation of Google Docs OCR, Tesseract, ABBYY FineReader, and Transym. In *International Symposium on Visual Computing*, pages 735–746. Springer.

Wei, T. C., Sheikh, U., and Ab Rahman, A. A.-H. (2018). Improved optical character recognition with deep neural network. In *2018 IEEE 14th International Colloquium on Signal Processing & Its Applications (CSPA)*, pages 245–249. IEEE.

Wen, Y., Zhang, K., Li, Z., and Qiao, Y. (2016). A discriminative feature learning approach for deep face recognition. In *European conference on computer vision*, pages 499–515. Springer.